IntechOpen

IntechOpen Series
Artificial Intelligence, Volume 15

Swarm Intelligence
Recent Advances and Current Applications

Edited by Marco Antonio Aceves-Fernández

Swarm Intelligence - Recent Advances and Current Applications

Edited by Marco Antonio Aceves-Fernández

Published in London, United Kingdom

Swarm Intelligence - Recent Advances and Current Applications
http://dx.doi.org/10.5772/intechopen.100661
Edited by Marco Antonio Aceves-Fernández

Contributors
Marco Antonio Aceves-Fernández, Marco Antonio Olguin-Sanchez, Jesus Carlos Pedraza-Ortega, Juan Manuel Ramos-Arreguín, Karim El Moutaouakil, Baïzri Hicham, Chellak Saliha, Cheggour Mouna, Rajesh Kumar Samala, Iraq T. Abbas, Saja Ayad, G. Devika, Asha Gowda Karegowda, Kazuaki Yamada, Pooria Mazaheri, Shahryar Rahnamayan, Azam Asilian Bidgoli, Soudamini Behera, Ajit Kumar Barisal, Sasmita Behera, Aslan Deniz Karaoglan, Deniz Perin, Kemal Yilmaz

First published in London, United Kingdom, 2023 by IntechOpen
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, 5 Princes Gate Court, London, SW7 2QJ, United Kingdom
Printed in Croatia

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Swarm Intelligence - Recent Advances and Current Applications
Edited by Marco Antonio Aceves-Fernández
p. cm.

This title is part of the Artificial Intelligence Book Series, Volume 15
Topic: Evolutionary Computation
Series Editor: Andries Engelbrecht
Topic Editor: Sebastian Ventura

Print ISBN 978-1-83768-086-3
Online ISBN 978-1-83768-087-0
eBook (PDF) ISBN 978-1-83768-088-7
ISSN 2633-1403

IntechOpen Book Series
Artificial Intelligence
Volume 15

Aims and Scope of the Series

Artificial Intelligence (AI) is a rapidly developing multidisciplinary research area that aims to solve increasingly complex problems. In today's highly integrated world, AI promises to become a robust and powerful means for obtaining solutions to previously unsolvable problems. This Series is intended for researchers and students alike interested in this fascinating field and its many applications.

Meet the Series Editor

Andries Engelbrecht received the Masters and Ph.D. degrees in Computer Science from the University of Stellenbosch, South Africa, in 1994 and 1999 respectively. He is currently appointed as the Voigt Chair in Data Science in the Department of Industrial Engineering, with a joint appointment as Professor in the Computer Science Division, Stellenbosch University. Prior to his appointment at Stellenbosch University, he has been at the University of Pretoria, Department of Computer Science (1998-2018), where he was appointed as South Africa Research Chair in Artifical Intelligence (2007-2018), the head of the Department of Computer Science (2008-2017), and Director of the Institute for Big Data and Data Science (2017-2018). In addition to a number of research articles, he has written two books, Computational Intelligence: An Introduction and Fundamentals of Computational Swarm Intelligence.

Meet the Volume Editor

Dr. Marco A. Aceves Fernández obtained his BSc (Eng) in Telematics from the Universidad de Colima, Mexico. He obtained both his MSc and Ph.D. in Intelligent Systems from the University of Liverpool, England. He is a full professor at the Universidad Autonoma de Queretaro, Mexico. He has been a member of the National System of Researchers (SNI) since 2009. He has published more than eighty research papers as well as several book chapters and congress papers. He has contributed to more than twenty funded research projects, both academic and industrial, in areas of artificial intelligence including environmental, biomedical, automotive, aviation, consumer, and robotics applications. He is also the honorary president of the National Association of Embedded Systems (AMESE), a senior member of the Institute of Electrical and Electronics Engineers (IEEE), a member of the Mexican Academia of Computer Science, and a board member of many institutions. His research interests include intelligent and embedded systems.

Contents

Preface

Since the earliest humans to populate the earth, we have gradually tried to understand and control the world around us. In trying to understand various phenomena, humans began to make predictions, for example, about the motions of the planets, eclipses, cycles of rainfall, or the periodicity of certain diseases. However, in the last few decades, the complexity of the predictions needed to be carried out has reached the limits of human ability. Luckily, the dawn of electronic computers has increased our abilities to predict nature, although the problems we are facing now are far more complex than the problems we faced a century ago.

Artificial intelligence (AI) machines demonstrate advanced cognitive skills in taking decisions, learning, perceiving the environment, predicting certain behavior, and processing written or spoken languages, among other skills. In addition, the ability of certain species to collectively demonstrate "swarm intelligence" has proven to be useful for solving complex problems that otherwise would be difficult to tackle. The collective power of AI and swarm intelligence can help solve difficult and complicated problems in innumerable areas.

This book highlights the wide range of applications in swarm intelligence and AI. It is divided into two sections: "Theoretical Background" and "Trends and Applications". It is a useful resource for researchers and students alike.

Marco Antonio Aceves-Fernández, Ph.D.
Universidad Autónoma de Querétaro,
Querétaro, México

Section 1

Theoretical Background

Chapter 1

Particle Swarm Optimization

Rajesh Kumar Samala

Abstract

The procedure is to obtain the best solution for the certain parameters in the given network to satisfy every requirement for design by considering the smallest affordable cost will be considered as an optimization. Optimization traditional approach will have some constraints like, outcome of single-based, local optima convergence, problems in unknown search space. To overcome the above constraints, many research organizations have established various metaheuristics to search optimization solutions for unsolved issues. The main intend to explain the Particle Swarm Optimization algorithm (PSOA) is to explain the stochastic optimization approach basics. Motivation of this Particle Swarm Optimization algorithm (PSOA) is to develop a strong metaheuristic optimization solution which is inspired natural swarm behavior like schooling of birds and fishes. PSOA is a simplified social network simulation. The final intent of this PSOA is a graphical representation and graphical simulate smoothly but undefined bird or fish flock's directions. Every bird's vicinity of observability is restricted to some area. Though having many birds permits every bird in the swarm fitness function to be bigger surface concerned. Mathematically every Particle Swarm Optimization algorithm (PSOA) has associated with fitness value, velocity, and position. Memory of maintaining global fitness, best position, and global fitness value.

Keywords: swarm optimization, position and velocity

1. Introduction

The optimization is having great impact on nature and human law of affairs. The characteristic of this optimization is to obtain the most superior (global minima or maxima). The optimization structure should maintain the generality. Hence the optimization design study should be obtaining the best of activity of human aspects and analyzing, understanding, solution and evaluating of mathematical issues of the system. This optimization issues are including some real problems constraints and these constraints must be satisfied to get the most feasible global solution. Many constraints are required in designing this optimization solution in real world practice.

The main objective this optimization is to obtain the best with the small effort. For example, in economic operation of electrical power system, with the minimum cost of operation obtaining the maximum efficiency. This is also applicable for many more fields in the real world.

2. Mathematical formulation

Generally, all the optimization design having the following steps:

i. Input data

ii. Initialization

iii. Constraints

iv. Evaluation

v. Updating

vi. Storing Result

The design of optimization will be expressed in a standard form as follows:

$$Objective\ Function = f(X) = maximizing\ or\ minimizing\ (cost) \tag{1}$$

Subjected to

$$\begin{gathered} Inequality\ Constraints\ for \\ minimizing\ objective\ function\ (pnumber)\ a_z(X) \leq 0,\ where\ z = 1, 2, 3,\ \dots, p \end{gathered} \tag{2}$$

$$\begin{gathered} Inequality\ Constraints\ for \\ maximizing\ objective\ function(p\ number)\ a_z(X) \geq 0,\ where\ z = 1, 2, 3,\ \dots, p \end{gathered} \tag{3}$$

$$Equality\ Constraints\ (q\ number)\ b_z(X) = 0,\ where\ z = 1, 2, 3,\ \dots, q \tag{4}$$

$$Input\ or\ design\ varibales\ defined\ as\ x_i,\ where\ i = 1, 2, 3,\ \dots, n \tag{5}$$

$$\text{Input or design variable expressed as } X = \left\{ \begin{matrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ x_n \end{matrix} \right\} \tag{6}$$

The design variables can also be expressed as $x_i^l = x_i = x_i^u$. Here x_i^l is the lower limit and x_i^u is the upper limit.

3. Present best and global best optimization

If design or Input variables that give the optimization like maximization or minimization of objective function *f(X)* can be represented with X^* (**Figure 1**).

Pints P1, P2, P3, P4, P5, P6 are known as Present best or Local best solutions and the P7 is known as the Global best solution for the given system.

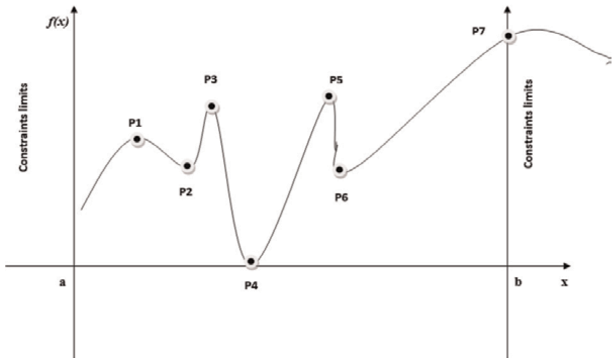


Figure 1.
Local minima and global minima [1].

Objective of this given system has a present or local best solution at X^* point, then $f\left(X^*\right) \leq f[X]$ for every X value and in single variable of x for a closed region of $a \leq x \leq b$.

4. Various optimization techniques

In the literature we have different types of mathematical approaches for optimization are found like Simplex approach, gradient approach, dynamic programming, branch and bound and integer programming etc. for all these approaches the size will be limited to solve. These approaches are more efficient in solving linear issues. With the increase in variables number and constraints number, the time of computation is also increases to obtain the solution. This increase in time increases exponentially. With the increase in complexity in solving complex issues with the help of mathematical approaches is also become more and more complex.

In last few years number of meta-heuristic approaches has developed to sole such complex issues with limited flexibility. Many numbers of bio and nature inspired optimization approaches has been developed like Genetic Algorithm (GA) and Swarm Intelligence (SI) approaches.

Here GA will work on the Darwinian principle for survival of the best solution in the population. GA uses the selection, cross-over and the mutation etc., as the operators to evaluate the population. Proved that, GA will give the closest solution for the global optimization with the introducing required improvements into it. Furthermore, the Differential Evaluation (DE) approach has explored to reach the global optimal solution.

Taking an Inspiration from living organism social behavior like fishes, birds, inspects etc. these can be communicate with each other might be directly or indirectly with a set of patterns used for the individual optimization. These approaches will operate on the cooperation of organisms instead of competing among them. All these exchange of information from one to other.

Here the Particle Swarm Optimization (PSO) is from the nature inspired behavior of food searching behavior of fishes and birds flocking behavior. To obtain the present best or local optimal solution and the global best or global optimal solution the fishes and the birds will be considered as particles.

As said previously PSO is the stochastic optimization method on swarm based. The finding the food is with the cooperative way in these swarms. Every particle (member) in this swarm ready to modify their pattern for search based on experience from learning from other particles (members) or by own.

Velocity and the position of particles will update in PSO based on the change in environment to reach proximity and the quality requirement. Here the proximity is the swarms must carry time and space calculations and the quality is swarms have to find the appropriate environmental change and have to respond.

According to cornfield model proposed by Heppner on the birds or fish flock behavior, let us assume a food location in the plane. Birds will randomly move or search for the food initially. Now (x_0, y_0) are the coordinates of position of cornfield. The individual birds coordinate for position and the velocity can be considered as (x, y) and (v_x, v_y) respectively. Distance from cornfield and the current position is used to determine the performance of the speed and the current position. If each particle has its own memory and can be able to memorize the best position it reached.

This present best position is represented with P_{best} (local best). Adjustment of velocity denoted with a constant '*a*', random numbers between [0, 1] denoted with '*rand*'. The below mathematical equations used to change the velocity as.

If $P_{best}\, x < x$, *then*
$V_x = V_x - (a * rand)$
else
$V_x = V_x + (a * rand)$
end
If $P_{best}\, y < y$
$V_y = V_y - (a * rand)$
else
$V_y = V_y + (a * rand)$
end

Now if there is a communication between swarms in some other way, every individual able to memorize their best location, considering global best position (gbest) of entire swarm.

Now if the constant for the velocity adjustment is denoted with 'b', by using above mathematical equations if the velocity adjusted, Gbest is also changed (or) updated as.

If gbest $< x$,
$V_x = V_x - (b * rand)$
else
$V_x = V_x + (b * rand)$
end.
If gbest $< y$
$V_y = V_y - (b * rand)$
else
$V_y = V_y + (b * rand)$
end

The final fixed velocity and position updating after some trial and errors as.

Velocity updating:

$$V_x = V_x + 2 * (pbestx - x) * rand + 2 * (gbestx - x) * rand \tag{7}$$

Position updating:

$$x = x + V_x \tag{8}$$

Here every individual is considered as a particle and these particles are not having any mass and volume. But these particles are having position as well as velocity. Hence this optimization is known as 'Particle Swarm Optimization Algorithm'.

This PSO algorithm can be explained using the following flowchart (**Figure 2**).

From the flowchart we assume that 'N' is the size of swarm. Position vector of every individual particle in D - dimensional vector space is

$$X_i = (X_{i1}, X_{i2}, X_{i3}, \dots, X_{id}, \dots, X_{iD}) \tag{9}$$

Velocity vector,

$$V_i = (V_{i1}, V_{i2}, V_{i3}, \dots, V_{id}, \dots, V_{iD}) \tag{10}$$

Present best (or) local best position of individual particle is,

$$P_i = (P_{i1}, P_{i2}, P_{i3}, \dots, P_{id}, \dots, P_{iD}) \tag{11}$$

Optimal (or) the best global position of individual particle

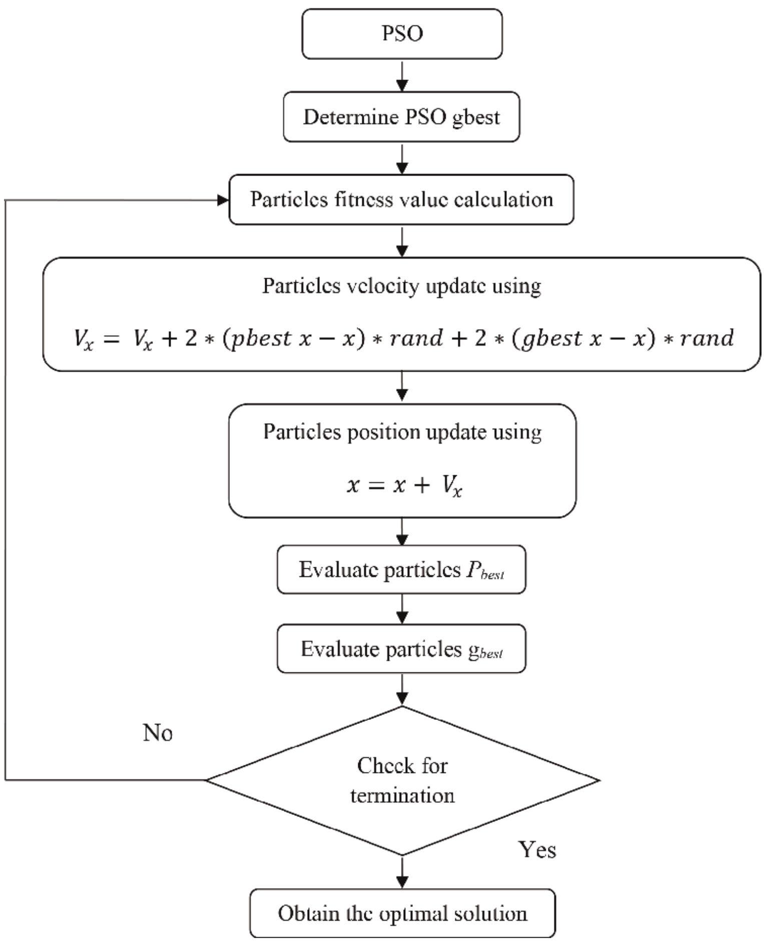


Figure 2.
The Flowchart of the PSOA [2].

$$P_g = \left(P_{g1}, P_{g2}, P_{g3}, \dots, P_{gd}, \ \dots, P_{gD}\right) \tag{12}$$

Now without generality loss, by considering minimization is an objective function, the mathematical equation for updating of individual particle optimal position as,

$if\, f\ (P_i) > f\ (x_i\ (t+1))$
$P_i^d, (t+1) = x_i^d$
else

$$P_i^d(t+1) = P_i^d$$

Now the velocity and position formula are given as,

$$V_i^d(t+1) = V_i^d(t) + C_1 * \left(P_i^d(t) - x_i^d(t)\right) * rand + C_1 * \left(P_i^d(t) - x_i^d(t)\right) * rand \tag{13}$$

$$x_i^d(t+1) = x_i^d(t) + V_i^d(t+1) \tag{14}$$

This proposed PSO was good enough for optimization. But later this PSO was modified to improve the effectiveness by adding Inertia weight. With Inertia weight introduction, the velocity update equation modified as,

$$V_i^d(t+1) = \omega * V_i^d(t) + C_1 * \left(P_i^d(t) - x_i^d(t)\right) * rand + C_1 * \left(P_i^d(t) - x_i^d(t)\right) * rand \tag{15}$$

Now by taking the convergence rate into an account, the construction factor (x) is introduced into PSO. Now the new update equation for velocity is became,

$$V_i^d(t+1) = x\left(V|i^d(t) + C_1 * \left(P_i^d(t) - x_i^d(t)\right) * rand + C_1 * \left(P_i^d(t) - x_i^d(t)\right) * rand\right) \tag{16}$$

Here the distance from the current position of the particle to its own best position called 'Cognitive' i.e., own thinking. Hence 'C_1' is called cognitive acceleration factor (or) cognitive learning factor.

Next the distance from current position of the particle to the global best position It is known as 'Social' factor. This indicates that there is a coordination and sharing of information between particles. Good particles movement through the cognition. Hence 'C_2' is the social acceleration factor (or) social learning factor.

Theory and practical application of this PSO algorithm have found great progress. This PSO used in various domains by every researcher with the understanding principle of operation and application.

PSO algorithm does not required any continuous derivative and differential optimized functions. This has fast rate of convergence. This PSO is very easy to understand, execute using programming.

Now coming to disadvantages: multiple local extreme functions, PSO may local minima and cannot produce optimal result because premature convergence of particle. PSO may not produce good results because of lack of best search techniques because of not using information sufficiently in procedure of calculations. In every Iteration PSO using only local optima Information which may not produce correct results. This PSO can be able to provide global search possibility but cannot guarantee for convergence to the global test. This PSO more suitable for optimization Issues s class with high dimensional and getting accurate outcome is not required because PSO is a meta-heuristic optimization algorithm. It never gives the correct explanation from the principle why it is efficient and not specified any of application range.

5. Various types of research about PSO

1. PSO algorithm can be analyzed theoretically and understanding of its working mechanism.

2. Structure change and obtain better performance.

3. Different parameters influence on PSO.

4. Different topology influence on PSO.

5. Parallel study of PSO algorithm.

6. Discrete study of PSO algorithm.

7. Multi-objective optimization study of PSO algorithm.

8. Use PSO algorithm for various fields in Engineering.

5.1 PSO for multi objective optimization

The single objective optimization is not correct practically Because the effective outcome is not accurate in single objective optimization. This problem has overcome by multi objective (MO) optimization. This MO has become the latest area in research. In this multi-objective optimization issues, all the target functions independently optimized and finally determine the best value for every target. But there is a conflict between objects. Because of conflicting between objects, unfortunately the finding of optimal solution is highly impossible for every objective hence only a pareto perfect solution has been determined.

Particles are independent agents in traditional PSO. Based on their own companion and its own experience problem space can be search by particles. As given earlier formula for particles update cognitive is the former and the social part is the latter. Here, selection of gbest and pbest (social and cognitive guide) is the important issue of Multi-Objective PSO (MOPSO). In both, i.e., in traditional PSO and MOPSO, choosing cognitive guide is same. But the guide must be found based on pareto dominance. There are two steps involved in choosing social guide.

Step-1: Candidate pool creation used for the guide selection.

In PSO traditional, this guide has been chosen from pbest (or) local (or) pbest of neighbors. While in case of MOPSO, to save more pareto optimal solutions, the normal technique is using an external pool.

Step 2: Guide Selection: Choosing of gbest must satisfy the below standards.

The chosen guide should be capable of provide particles guidance effectively. This is because of improve the speed of convergence.

The selected guide required to give balanced search with pareto frontier. This is because to maintain population diversity.

To select social guide here two methods are specified.

1. Quantity standard

2. Roulette mode of selection

3. Quantity standard: Selection of social guide by following various procedures, but not random selection involvement.

4. Roulette mode of selection: Random selection based on various standards, in order to maintain population diversity.

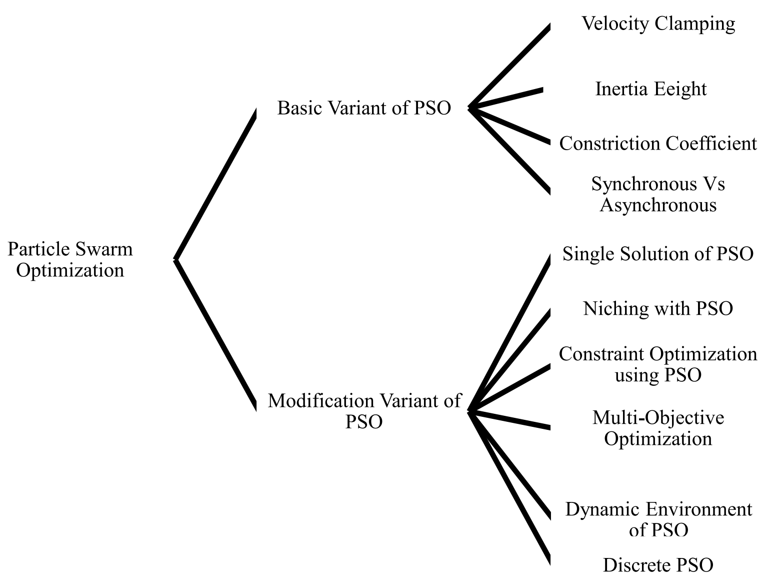
Pseudo code for PSO algorithm:

Initialize population
for *t = 1: maximum generation*
  for *i = 1: population size*
    if $f(x_{i,d}(t)) < f(p_i(t))$ *then* $p_i(t) = x_{i,d}(t)$
      $f(p_g(t)) = min(f(p_i(t)))$
    end
    for *d = 1: dimension*
    $v_{i,d}(t+1) = w_{vi,d}(t) + C_{1*} r_1(p_i - x_{i,d}(t)) + C_{2*} r_2(p_g - x_{i,d}(t))$
    $x_{i,d}(t+1) = x_{i,d}(t) + v_{i,d}(t+1)$
    if $v_{i,d}(t+1) > v_{max}$ *then* $v_{i,d}(t+1) = v_{max}$
    else if $v_{i,d}(t+1) < v_{min}$ *then* $v_{i,d}(t+1) = v_{min}$
    end
    if $x_{i,d}(t+1) > x_{max}$ *then* $x_{i}, d(t+1) = x_{max}$
    else if $x_{i,d}(t+1) < x_{min}$ *then* $x_{i,d}(t+1) = x_{min}$
    end
  end
  end
end
For each particle
    Initialize particle
END
Do
  For each particle
      Calculate fitness value
      If the fitness value is better than the best fitness value (pbest) in history set current value
        as the new pbest
  End
Choose the particle with the best fitness value of all the particles as the gbest
For each particle
Calculate particle velocity
Update particle position
End

While maximum iterations or minimum error criteria is not attained (**Table 1**).

PSO Basic Variant	Function	Advantages	Disadvantages
Velocity Clamping (VC)	Control the global exploration of the particle. Reduces the size of the step velocity, so that the particles remain in the search area, but it cannot change the search direction of the particle.	VC reduces the size of the step velocity so it will control the movement of the particle	If all the velocity becomes equal to the particle will continue to conduct searches within a hypercube and will probably remain in the optima but will not converge in the local area.
Inertia Weight	Controls the momentum of the particle by weighing the contribution of the previous velocity	A larger inertia weight in the end of search will foster the convergence ability.	Achieve optimally convergence strongly influenced by the inertia weight
Construction Coefficient	To ensure the stable convergence of the PSO algorithm	Similar with inertia weight	When the algorithm converges, the fixed values of the parameters might cause the unnecessary fluctuation of particles
Synchronous and Asynchronous Updates	Optimization in parallel processing	Improved convergence rate	Higher throughput: More sophisticated finite element formulations Higher accuracy (mesh densities)

Table 1.
The basic variant of PSO [3].



The PSO algorithm can be hybridized with several metaheuristic algorithm to balance the exploration and exploitation. Some algorithms have better efficiency in exploration, but they are poor in exploitation. Some algorithms will take high iteration to reach convergence and show poor performance. In this case the PSO is introduced to enhance the performance using,

$$V_i(K+1) = \omega V_i(K) + \rho_i * C_1\left(P_i^{best} - X_i(K)\right) + \rho_2 * C_2(GX_i(K)) \tag{17}$$

$$X_i(K+1) = X_i(K) + V_i(K+1) \tag{18}$$

The proposed algorithm using PSO is written as,

1. Define population size (P), Initial values, C_1 and C_2 as acceleration constants, P_c the number of variables, and maximum iteration number as Max_{iter}.

2. Setting $t = 0$, as counter initialization.

3. for $i = 1$: *if* $P \geq i$, do

4. Generate an initial random population $X_i(t)$.

5. Fitness evaluation function of every search agent or solution $f(X_i)$.

6. end for

7. repeat

8. Apply standard original Particle Swarm Optimization (PSO) Algorithm.

9. Apply the selection operate of another algorithm which are going to hybridize with PSO. (Ex: Genetic Algorithm (GA), Ant-Lion Optimization Algorithm (ALOA) etc.)

10. Partition the population $X(t)$ into in to $part_{no}$ sub-partitions, where each sub-partition $X'(t)$ size is $v \times \eta$.

11. for $i = 1$: $part_{no} \geq i$ do

12. Apply the arithmetical crossover as shown in procedure 1 on each sub-partition $X'(t)$.

13. end for

14. Apply the algorithm which is hybridized with PSO on the whole population $X(t)$.

15. Update the solution in the population $X(t)$.

16. Set t = t + 1. Increase the iteration count.

17. Until iteration reaches the Max_{iter}. Satisfied the termination criteria.

18. Print the best solution (**Figure 3**).

PSOA will be applied in various optimization areas, example, Energy-Storage Optimization, Image Processing, Economic operation of power system, Optimal location identification, analysis of slope stability, foundation and pile engineering, soil, and rock mechanics, underground and tunneling space design etc. PSOA will simulate the particle movement and will be applied in visual effects and special effects in some films.

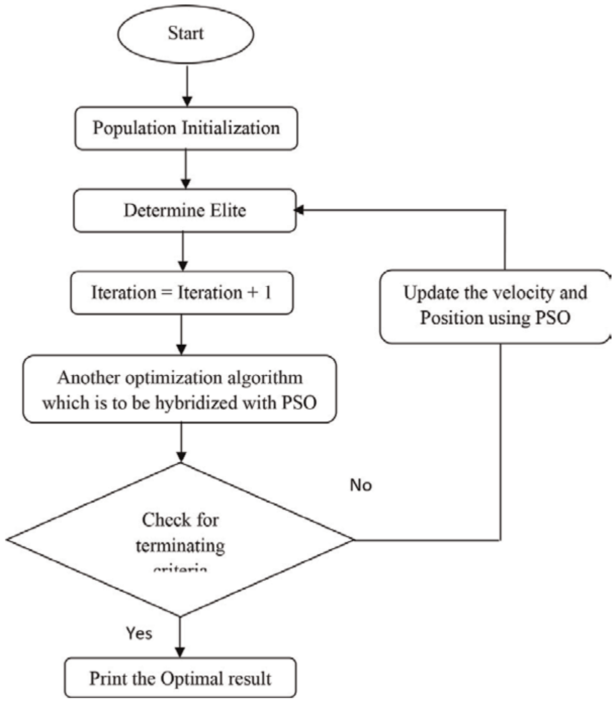


Figure 3.
The flowchart of the hybridization with PSOA [2].

Along with the conventional applications optimization problems, Swarm Intelligence (SI) will be used in acquisition of material from library, communications, classification of medical dataset, dynamic control, planning of heating system, tracking of moving objects, and prediction.

PSO Algorithm Visual Explanation:

Let us suppose a group of birds are flying randomly in an area in search of their food (**Figure 4**).

No bird knows exactly where the food is, but they know how far they are in each iteration, (**Figure 5**).

Birds they do not know the best position. If any member can find the desirable path to go, the rest of the members will follow quickly (**Figure 6**).

Finally, following the bird which is nearest to the food is an aim.

Figure 4.
Random movement of birds [4].

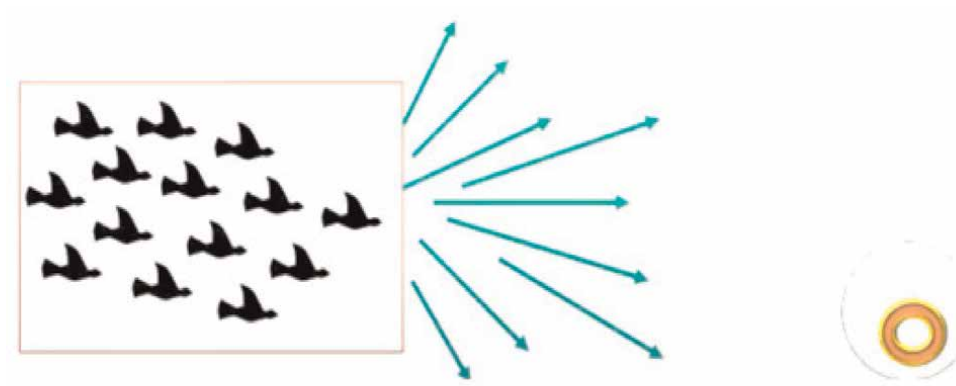


Figure 5.
Birds moving toward the food [4].

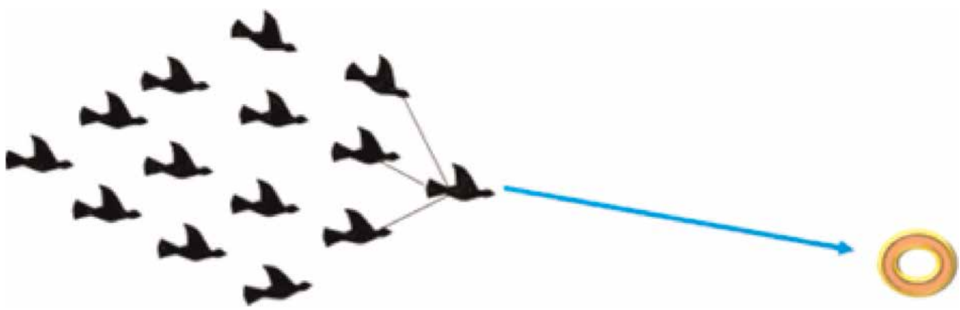


Figure 6.
Finding best path [4].

5.2 PSO algorithm demo with one example

Objective function used for Minimization:

$$Fitness = 10 * (X_1 - 1)\hat{}2 + 20 * (X_2 - 2)\hat{}2 + 30 * (X_3 - 3)\hat{}2$$

STEP – 1: PARAMETERS INITIALIZATION (Table 2):

Number of variables: *m = 3.*

Population size: *n = 5.*

Inertia weight: $Wmax = 0.9$, $W_{min} = 0.4$.

8	**9**	**1**
9	**6**	**1**
3	**5**	**10**
10	**2**	**10**
10	**5**	**8**

Table 2.
Output [5].

Acceleration Factor: $C_1 = 2, C_2 = 2.$
Maximum Iteration: $Max_{iter} = 50.$

% Population Initialization

LB = [0 0 0];
UB = [10 10 10];

```
for i = 1: n
    for j = 1:m
    X0 (i, j) = round (LB (j) + rand () * (UB (j) – LB (j)));
    end
end
```

- **Initialize velocity (v_i) randomly for each particle: (Table 3)**

- $v = 0.1 * X_0 (i, j);$

Initial velocity for First Particle:
$v(X_1) = 0.1 * 8 = 0.8$
$v(X_3) = 0.1 * 9 = 0.9$
$v(X_1) = 0.1 * 1 = 0.1$
Initial velocity for Second Particle:
$v(X_1) = 0.1 * 9 = 0.9$
$v(X_3) = 0.1 * 6 = 0.6$
$v(X_1) = 0.1 * 1 = 0.1$
Initial velocity for Third Particle:
$v(X_1) = 0.1 * 3 = 0.3$
$v(X_3) = 0.1 * 5 = 0.5$
$v(X_1) = 0.1 * 10 = 1.0$
Initial velocity for Fourth Particle:
$v(X_1) = 0.1 * 10 = 1.0$
$v(X_3) = 0.1 * 2 = 0.22$
$v(X_1) = 0.1 * 10 = 1.0$
Initial velocity for Fifth Particle:
$v(X_1) = 0.1 * 10 = 1.0$
$v(X_3) = 0.1 * 5 = 0.5$
$v(X_1) = 0.1 * 8 = 0.8$
Initial Position (Xi) Randomly for each Particles **(Table 4)*****:***
Current Position = Previous Position + Velocity
Current Position for First Particle:
$X_1 = 8 + 0.8 = 8.8$

	X_1	X_2	X_3
1st Particle	8	9	1
2nd Particle	9	6	1
3rd Particle	3	5	10
4th Particle	10	2	10
5th Particle	10	5	8

Table 3.
Initial population [5].

	X_1	X_2	X_3
1st Particle	***0.8***	***0.9***	***0.1***
2nd Particle	***0.9***	***0.6***	***0.1***
3rd Particle	***0.3***	***0.5***	***1.0***
4th Particle	***1.0***	***0.2***	***1.0***
5th Particle	***1.0***	***0.5***	***0.8***

Table 4.
Initial velocity [5].

X_2 = 9 + 0.9 = 9.9
X_3 = 1 + 0.1 = 1.1
Current Position for Second Particle:
X_1 = 9 + 0.9 = 9.9
X_2 = 6 + 0.6 = 6.6
X_3 = 1 + 0.1 = 1.1
Current Position for Third Particle:
X_1 = 3 + 0.3 = 3.3
X_2 = 5 + 0.5 = 5.5
X_3 = 10 + 1 = 11
Current Position for Fourth Particle:
X_1 = 10 + 1 = 11
X_2 = 2 + 0.2 = 2.2
X_3 = 10 + 1 = 11
Current Position for Fifth Particle:
X_1 = 10 + 1 = 11
X_2 = 5 + 0.5 = 5.5
X_3 = 8 + 0.8 = 8.8
STEP – 2: FITNESS EVALUATION $f\left(x_i^t\right)$ **(Table 5)**
Fitness Evaluation for Every Particle:
The objective function taken for minimization:

$$F(x) = 10 * (X_1 - 1)\hat{}2 + 20 * (X_2 - 2)\hat{}2 + 30 * (X_3 - 3)\hat{}2$$

$$F\left(x_1^0\right) = 10 * (8.8 - 1)\hat{}2 + 20 * (9.9 - 2)\hat{}2 + 30 * (1.1 - 3)\hat{}2$$

$$F\left(x_1^0\right) = 1.9649$$

	X_1	X_2	X_3
1st Particle	***8.8***	***9.9***	***1.1***
2nd Particle	***9.9***	***6.6***	***1.1***
3rd Particle	***3.3***	***5.5***	***11***
4th Particle	***11***	***2.2***	***11***
5th Particle	***11***	***5.5***	***8.8***

Table 5.
Current position [5].

$$F(x_2^0) = 10 * (9.9 - 1)^\wedge 2 + 20 * (6.6 - 2)^\wedge 2 + 30 * (1.1 - 3)^\wedge 2$$

$$F(x_2^0) = 1.3236$$

$$F(x_3^0) = 10 * (3.3 - 1)^\wedge 2 + 20 * (5.5 - 2)^\wedge 2 + 30 * (11 - 3)^\wedge 2$$

$$F(x_3^0) = 2.2179$$

$$F(x_4^0) = 10 * (11 - 1)^\wedge 2 + 20 * (2.2 - 2)^\wedge 2 + 30 * (11 - 3)^\wedge 2$$

$$F(x_4^0) = 2.9208$$

$$F(x_5^0) = 10 * (11 - 1)^\wedge 2 + 20 * (5.5 - 2)^\wedge 2 + 30 * (8.8 - 3)^\wedge 2$$

$$F(x_5^0) = 2.2542$$

g_{Best} = 1.3236 **(Table 6)**

STEP – 3: CALCULATION OF POSITION AND VELOCITY FOR EVERY PARTICLE

- ***Particle Position Calculation:*** $x_i^{t+1} = x_i^t + v_i^t$
- ***Particle Velocity Calculation:***

$$v_i^{t+1} = wv_i^t + C_1 r_1 \left(xBest_i^t - x_i^t\right) + C_2 r_2 \left(gBest_i^t - x_i^t\right)$$

Velocity Update for at iteration (t = 0) for First Particle:

$$v_1^{0+1} = wv_1^0 + C_1 r_1 \left(xBest_1^0 - x_1^0\right) + C_2 r_2 \left(gBest_1^0 - x_1^0\right)$$

$$v_1^1 = 0.9 * 0.8 + 2 * rand() * (8.8 - 8.8) + 2 * rand() * (9.9 - 8.8)$$

$$v_1^1 = 1.0667$$

Calculating all the values as:

$$v_1^1 = 1.0667 \, for \, x_1$$

$$v_1^1 = -4.4719 \, for \, x_2$$

$$v_1^1 = 0.0900 \, for \, x_3$$

FITNESS VALUE (t = 0)
1.9649
1.3236
2.2179
2.9208
2.2542

Table 6.
Choose the particle with best fitness value as g_{Best} [5].

	X_1	X_2	X_3
1st Particle	1.0667	−4.4719	0.0900
2nd Particle	0.8100	0.5400	0.0900
3rd Particle	7.4888	2.5728	−18.3177
4th Particle	−0.1678	1.4286	1.4286
5th Particle	−1.2109	0.5286	−13.6635

Table 7.
Updated velocity for every particle [5].

Particle Position Update: **(Table 7)**
Position Update for First Particle:

$$x_1^{0+1} = x_1^0 + v_1^0$$

$$x_1^1 = 8.8 + 1.0667 = 9.8667$$

$$x_1^1 = 9.9 + (-4.4719) = 5.4281$$

$$x_1^1 = 1.1 + 0.900 = 2$$

STEP – 4: FITNESS EVALUATION $f\left(x_i^t\right)$ **(Table 8)**
Current Best Calculation Best [gBest]
Update Minimization: Calculate Fitness Value:

$$F(x) = 10 * (X_1 - 1)\hat{}2 + 20 * (X_2 - 2)\hat{}2 + 30 * (X_3 - 3)\hat{}2$$

$$FirstParticle$$

$$F(x) = 10 * (9.8667 - 1)\hat{}2 + 20 * (5.4281 - 2)\hat{}2 + 30 * (2 - 3)\hat{}2 = 1.0512$$

$$SecondParticle$$

$$F(x) = 10 * (10.71 - 1)\hat{}2 + 20 * (7.14 - 2)\hat{}2 + 30 * (1.19 - 3)\hat{}2 = 1.5695$$

$$ThirdParticle$$

$$F(x) = 10 * (10.78 - 1)\hat{}2 + 20 * (8.07 - 2)\hat{}2 + 30 * (-7.317 - 3)\hat{}2 = 4.8866$$

$$FourthParticle$$

$$F(x) = 10 * (10.78 - 1)\hat{}2 + 20 * (8.07 - 2)\hat{}2 + 30 * (-7.317 - 3)\hat{}2 = 3.6716$$

	X_1	X_2	X_3
1st Particle	9.8667	5.4281	2
2nd Particle	10.71	7.14	1.19
3rd Particle	10.83	8.07	−7.317
4th Particle	10.83	3.628	12.42
5th Particle	9.78	6.028	−4.8635

Table 8.
Updated positions for every particle [5].

Updated Fitness Value
1.0512
1.5695
4.8866
3.6716
2.9504

Table 9.
Updated fitness value [5].

$$FifthParticle$$
$$F(x) = 10 * (9.78 - 1)\hat{}2 + 20 * (6.028 - 2)\hat{}2 + 30 * (-4.8635 - 3)\hat{}2 = 2.9504$$

Choose the Current Best as **(Table 9)**
$\boldsymbol{if}\, f(x_1^1) < f(gBest) then$
$gBest = x_1^1$
$\boldsymbol{if}\, f(x_1^1) < f(gBest)$
1.0512 < 1.3236
$\boldsymbol{if}\, f(x_1^1) < f(gBest)\ \ then$
$gBest = 1.0512$
$\boldsymbol{if}\, f(x_2^1) < f(gBest)$

1.5695 < 1.3236
$\boldsymbol{if}\, f(x_3^1) < f(gBest)$
4.8866 < 1.3236
$\boldsymbol{if}\, f(x_4^1) < f(gBest)$
3.6716 < 1.3236
$\boldsymbol{if}\, f(x_5^1) < f(gBest)$
2.9504 < 1.3236
gBest = 1.0512
$\boldsymbol{if}\, f(x_i^t) < f(pBest)\ then$
pBest = x_i^t
$if\, f(x_1^1) < f(pBest)$
1.0512 < 1.9649
$\boldsymbol{if}\, f(x_2^1) < f(pBest)$
1.5695 < 1.3236
$\boldsymbol{if}\, f(x_3^1) < f(pBest)$
4.8866 < 2.2179
$\boldsymbol{if}\, f(x_4^1) < f(pBest)$
3.6716 < 2.9208
$\boldsymbol{if}\, f(x_5^1) < f(pBest)$
2.9504 < 2.2542
STEP – 5: UPDATE ITERATION (Table 10)
Update t = t + 1 = 2
t = 2

New pBest Value
1.0512
1.3236
7.2179
2.9208
2.2542

Table 10.
Updated fitness value [5].

1st Particle	9.8667	5.4281	2

Table 11.
OUTPUT gBest & x_i^t [5].

STEP – 6:
gBest = 1.0512
Repeat this until stopping Criteria met **(Table 11)**

5.3 PSO algorithm demo with one more example

Find the minimum of the function using PSO algorithm.

$$f(x) = -x^2 + 5x + 20$$

Using 9 particles with initial positions.

$$x_1 = -9.6, x_2 = -6, x_3 = -2.6, x_4 = -1.1, x_5 = 0.6, x_6 = 2.3, x_7 = 2.8, x_8 = 8.3, x_9 = -10,$$

Step 1: Evaluation of objective function as

$$f_1^0 = -120.16, f_2^0 = -46, f_3^0 = 0.24, f_4^0 = 13.29, f_5^0 = 22.64, f_6^0 = 26.21, f_7^0 = 26.16, f_8^0 = -7.39, f_9^0 = -30$$

Let C1 = C2 = 1 and set initial velocities of the particles to zero.

$$v_1^0 = 0, v_1^0 = v_2^0, v_3^0, v_4^0, v_5^0, v_6^0, v_7^0, v_8^0, v_9^0 = 0$$

Step 2: Set the iteration number as t = 0 + 1 and go to step 3.
Step 3: Find the P_{best} for every particle.

$$P_{best,\ i}^{t+1} = \begin{cases} P_{best,\ i}^{t} iff_i^{t+1} > P_{best,\ i}^{t} \\ x_i^{t+1} iff_i^{t+1} \le P_{best,\ i}^{t} \end{cases}$$

So,

$$P^1_{best,\,1} = -9.6, P^1_{best,\,2} = -6, P^1_{best,\,3} = -2.6, P^1_{best,\,4} = -1.1, P^1_{best,\,5} = 0.6,$$
$$P^1_{best,\,6} = 2.3, P^1_{best,\,7} = 2.8, P^1_{best,\,8} = 8.3, P^1_{best,\,9} = 10$$

Step 4: Gbest = max(Pbest) so Gbest = 2.3.

Step 5: Updating the velocities of the particle by considering the value of random numbers r1 =0.213, r2 = 0.876, C1 = C2 = 1, w = 1.

$$v_i^{t+1} = v_i^t + C1r_1^t\left[P^t_{best,\,i} - x_i^t\right] + C2r_2^t\left[G^t_{best} - x_i^t\right]; \quad i = 1, 2, 3, \ \dots, \ 9.$$
$$v_1^1 = 0 + 0.213(-9.6 + 9.6) + 0.876(2.3 + 9.6) = 10.4244$$
$$v_2^1 = 7.2708, v_3^1 = 4.2924, v_5^1 = 1.4892, v_6^1 = 0, v_7^1 = -0.4380, v_8^1 = 5.256, v_9^1 = -6.7452$$

Step 6: Update the values of position as well.

$$x_i^{t+1} = x_i^t + v_i^{t+1}$$
$$x_1^1 = 0.8244, x_2^1 = 1.2708, x_3^1 = 1.6924, x_4^1 = 1.8784, x_5^1 = 2.0892, x_6^1 = 2.3,$$
$$x_7^1 = 2.362, x_8^1 = 3.044, x_9^1 = 3.2548$$

Step 7: Finding objective function values of

$$f_1^1 = 23.4424, f_2^1 = 24.739, f_3^1 = 25.5978, f_4^1 = 25.8636, f_5^1 = 26.0812, f_6^1 = 26.21,$$
$$f_7^1 = 26.231, f_8^1 = 25.9541, f_9^1 = 25.6803$$

Step 8: Stopping Criteria.

If the terminal rule is satisfied, go to step 2. Otherwise stop the iteration and note the result.

PSO Algorithm Demo with nature inspired hybrid algorithm:

The optimal integration of Distributed Generation (DG) would be investigated by determining the reduced power loss, enhanced voltage at every bus, reduced total operating cost and enhanced voltage stability of the test network.

In this section, the candidate busses for DG placement and the capacity of the DGs are estimated with the help of Ant-Lion Optimization Algorithm (ALOA) methodology.

Later the elitism phase of the ALOA methodology for updating the DGs location will be done by introducing Particle Swam Optimization Algorithm (PSOA) in ALOA.

The ALOA imitates the ant-lion's hunting mechanism. In ant-lion's life cycle there are two main stages, namely, Larva stage or phase and adult stage. The antlion using its larva stage to find the food and the adult stage for reproduction. The inspiration for this ALOA is the larva stage. The antlion prepare one pit by digging in the sand in the shape of cone by moving in a circular way and throwing the sand out of the pit with its heavy jaw. After preparation of trap the antlion will wait for food. The level of hungry of an antlion and the size of the moon decides the trap size. If any pray come toward the surface of the trap, that will fall into the trap very easily. Then the antlion will catch the pray, when it is identified by the antlion.

The matrices $M_{ant\text{-}lion}$ and M_{ant} stores the random places of the antlions and ants respectively,

$$M_{ant-lion} = \begin{pmatrix} AL_{1,\,1} & AL_{1,\,2} & \dots & AL_{1,\,n} \\ AL_{2,\,1} & AL_{2,\,2} & \dots & AL_{2,\,d} \\ \vdots & \vdots & \vdots & \vdots \\ AL_{n,\,1} & AL_{n,\,2} & \dots & AL_{n,\,d} \end{pmatrix}$$
$$M_{ant} = \begin{pmatrix} A_{1,\,1} & A_{1,\,2} & \dots & A_{1,\,n} \\ A_{2,\,1} & A_{2,\,2} & \dots & A_{2,\,d} \\ \vdots & \vdots & \vdots & \vdots \\ A_{n,\,1} & A_{n,\,2} & \dots & A_{n,\,d} \end{pmatrix} \tag{19}$$

Antlion's and ant's fitness will be stored in $M_{fit\text{-}AL}$ and $M_{fit\text{-}A}$ matrices,

$$M_{fit-AL} = \begin{pmatrix} f([AL_{1,\,1}, & AL_{1,\,2}, & \dots, & AL_{1,\,n}]) \\ f([AL_{2,\,1}, & AL_{2,\,2}, & \dots, & AL_{2,\,d}]) \\ & & \vdots & \\ f([AL_{n,\,1}, & AL_{n,\,2}, & \dots, & AL_{n,\,d}]) \end{pmatrix}$$
$$M_{fit-A} = \begin{pmatrix} f([A_{1,\,1}, & A_{1,\,2}, & \dots, & A_{1,\,n}]) \\ f([A_{2,\,1}, & A_{2,\,2}, & \dots, & A_{2,\,d}]) \\ & & \vdots & \\ f([A_{n,\,1}, & A_{n,\,2}, & \dots, & A_{n,\,d}]) \end{pmatrix} \tag{20}$$

5.3.1 Ant's random movement

When ants are moving randomly in search space, then the antlions are ready to find them. All the ants randomly move in search space to find their food is given as,

$$\begin{aligned} ant\ position = [0, cumulative\ sum(2^{*} random\ number - 1\,),\ \dots \\ till\ \max imum\ iteration] \end{aligned} \tag{21}$$

$$random\ number = \begin{cases} 1\ if\ rand > 0.5 \\ 0\ if\ rand \le 0.5 \end{cases} \tag{22}$$

The random movement of ants is restricted within search space. Hence, the ant's position will be updated as,

$$New\ ant\ position \ = \frac{(old\ ant\ position - a_m)\,\left(d_m - c_m^t\right)}{b_m - a_m} + c_t \tag{23}$$

Where, a_m and b_m represents minimum and maximum of ant's random walk, ${c^t}_m$ and ${d^t}_m$ represents minimum and maximum of m^{th} variable at t^{th} iteration.

5.3.2 Trapping of ants

The equation for the ant's trapping into ant-lion's pit is as,

$$\begin{aligned} c_m^t &= Ant - lion_n^t + c^t \\ d_m^t &= Ant - lion_n^t + d^t \end{aligned} \tag{24}$$

Construction of trap

By using roulette wheel method, the best ant-lion is chosen.

Ants sliding toward ant lion

Ants sliding into the pit are represented by,

$$\begin{aligned} c^t &= \frac{c^t}{10^{\frac{wt}{s}}} \\ d^t &= \frac{d^t}{10^{\frac{wt}{s}}} \end{aligned} \tag{25}$$

$$w = \begin{cases} 2 \ \ if\, t > 0.1S \\ 3 \ \ if\, t > 0.5S \\ 4 \ \ if\, t > 0.75S \\ 5 \ \ if\, t > 0.9S \\ 6 \ \ if\, t > 0.95S \end{cases} \tag{26}$$

5.3.3 Catching the pray and reconstruction process of pit

If the ant's objective function is better than that of the antlions, then the ant-lion moved to a new position to find the opportunity to catch the pray in a better way. The new ant-lion's position is given as,

$$\begin{aligned} Ant - lion\ new\ position = Ant\ position\ \ for\, ant\ objective\ funtion > \\ ant - lion\ objective\ function \end{aligned} \tag{27}$$

5.3.4 Elitism

In each stage the best solution is stored called Elitism. Each ant is assumed to be connected to an antlion using roulette wheel and is represented by,

$$elite\ ant = \frac{random\ walk\ around\ the\ selected\ ant - lion}{2} \tag{28}$$

5.3.5 The particle swarm optimization algorithm (PSOA)

This optimization algorithm is stochastic population-based method to solve optimization issues. PSO is a nature inspired meta-heuristic algorithm, works with a particles swarm. In a search space every particle is considered as a solution with two characteristics namely, its own velocity and position. The velocity represents the direction and the distance for optimize the position in next iteration. The position defines the present value of the solution. For the entire particle its own present best named as *pbest* and the overall global solution named as *Gbest* estimated.

5.3.6 Mathematical formulation for PSOA

PSO is a stochastic population-based metaheuristic to solve continuous optimization problems. The main idea of the metaheuristic came from the observation of behavior of natural organisms to find food. PSO works with a swarm of particles. Each particle is a solution to a problem in the decision space and has two characteristics: its own position and velocity. The position represents the current values in the solution; the velocity defines the direction and the distance to optimize the position at next iteration. For each particle i its own past best position p best i and the entire swarm's best overall position G are remembered. In basic PSO the velocity and position of each particle are updated in the following equations,

$$v_i(k+1) = wv_i(k) + \rho_1 c_1\left(p_i^{best} - X_i(k)\right) + \rho_2 c_2(GX_i(k)) \tag{29}$$

$$X_i(k+1) = X_i(k) + v_i(k+1) \tag{30}$$

Where, i is a particle index, k is an iteration number, $v_i(k)$ is velocity, $X_i(k)$ is the position of particle i at iteration k, p_i^{best} is the best position found by particle i (personal best), G is the best position found by the swarm (global best, best of personal bests), w is an inertia coefficient, (ρ_1, ρ_2) are random numbers in [0,1] interval, c_1, c_2 are positive constants representing the factors of particle's attraction toward its own best position or toward the swarm's best position (**Figure 7**).

5.3.7 Hybridization of particle swarm optimization algorithm (PSOA) based ant-lion optimization algorithm (ALOA)

In this part of hybridization is, the PSOA has been associated to enhance the ALOA performance, by updating the ALOA elitism stage.

Initially the load flow analysis is carried out with the help of Backward/Forward (BW/FW) sweep method for the base case with the bus data and the load data of the test system without adding any extra load on to the system. Now with intend to reduce these power losses up to maximum possible extent the PSOA based ALOA is combined to form a hybrid methodology. With this methodology, the candidate busses, and capacities of DGs are estimated with the help of ALOA method, and then the locations of antlions are updated with the help of PSOA methodology. This will be evaluated with the help of the flowchart given for proposed hybrid methodology.

5.3.8 Specific contribution

Initially by using the ALOA standalone algorithm the DG locations and capacities were identified intend to reduce the total power losses and total operating cost and enhancement of voltage stability index and the voltage profile at every bus. All the values have been tabulated with different power factor conditions on various test networks with different types of DG units.

Now with the help of proposed hybrid algorithm the DG locations and capacities were identified intend to reduce the total power losses and total operating cost and enhancement of voltage stability index and the voltage profile at every bus. All the values have been tabulated with different power factor conditions on various test networks with different types of DG units.

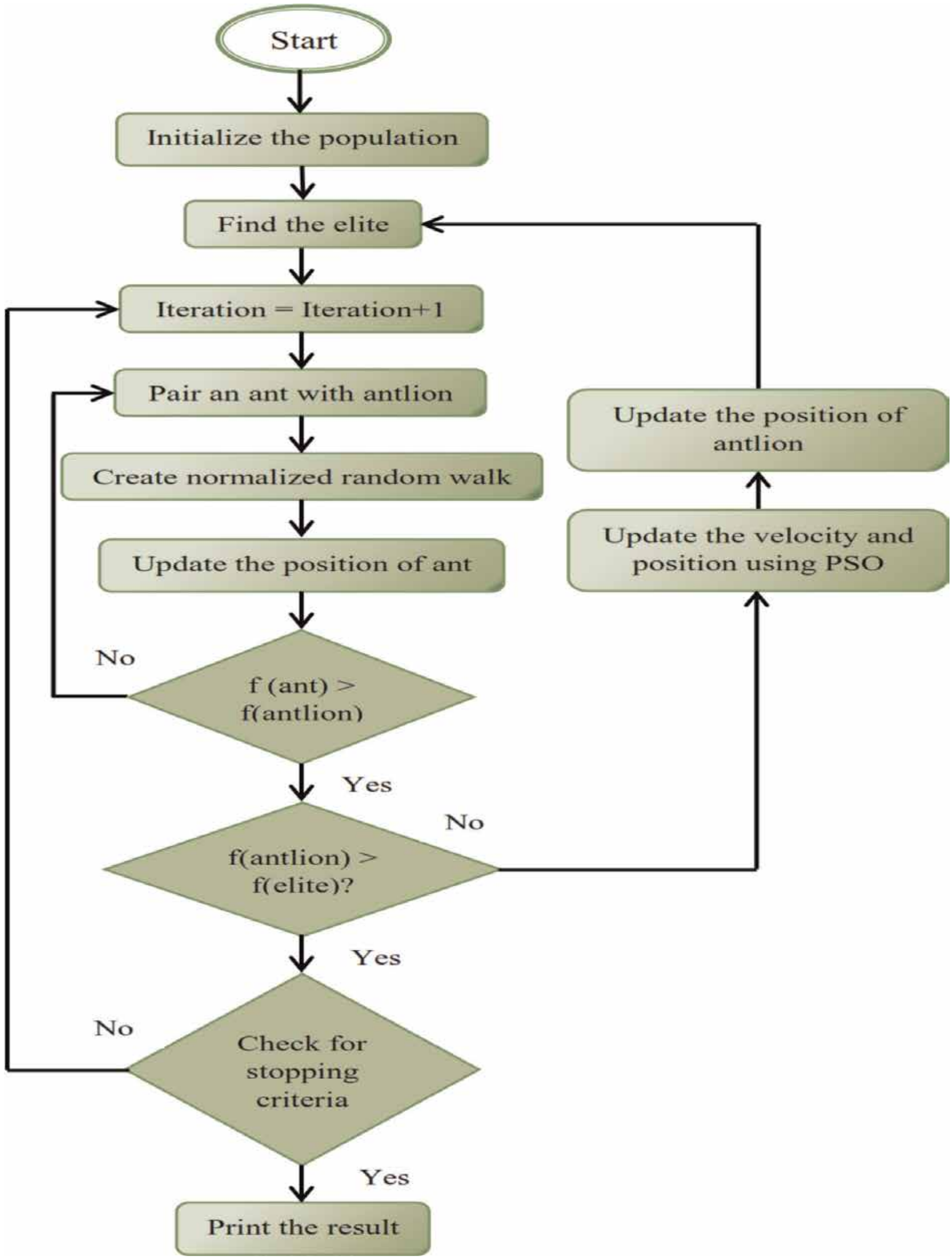


Figure 7.
Flow chart of proposed PSOA-ALOA hybrid technique [6].

In ALOA hybridized with PSOA the elitism phase of the ALOA for DG position were updated by using the following mathematical equations,

$$\begin{aligned} &new\ volocity_j\ (kk+1) = cons\tan t^*\ old\ velocity_j\ (kk) \\ &+ inertia\ cons\tan t_{1*} positive\ cons\tan t_1{}^*\ (p_j^{best} - position\ (kk) \\ &+ inertia\ cons\tan t_{2*} positive\ cons\tan t_2{}^*\ \left(G_j^{best} - position\ (kk)\right) \end{aligned} \tag{31}$$

$$new\ position_j\ (kk+1) = old\ position_j\ (kk) + new\ velocity\ (kk+1) \quad (32)$$

This is because the performance of the ALOA has high iteration to reach the optimal solution and poor performance. To improve the performance of ALOA is based on the PSOA. This property leads to this hybridization of ALOA-PSOA for optimal integration of DG units were identified with intend to reduce the total power losses and total operating cost and enhancement of voltage stability index and the voltage profile at every bus. All the values have been tabulated with different power factor conditions on various test networks with different types of DG units.

The PSOA has still having some issues that ought to be resolved. Therefore, the future works on the PSOA will probably concentrate on the following:

1. Find a particular PSO algorithm which can be expected to provide good performance.

2. Combine the PSO algorithm with other optimization methods to improve the accuracy.

3. Use this algorithm to solve the non-convex optimization problems.

Author details

Rajesh Kumar Samala
Department of Electrical and Electronics Engineering, Anurag University, Hyderabad, India

*Address all correspondence to: rajeshkumar_samala@yahoo.co.in

References

[1] Kennedy J, Eberhart RC. Particle swarm optimization. In: Proceedings of the International Conference on Neural Networks; Institute of Electrical and Electronics Engineers. Vol. 4. 1995. pp. 1942–1948. DOI: 10.1109/ICNN.1995.488968

[2] Samala RK, Kotapuri MR. Distributed generation allocation in distribution system using particle swarm optimization based ant-lion optimization. International Journal of Control and Automation. 2020;**13**(1)

[3] Eberhart RC, Shi Y. Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation (CEC'00). Vol. 1. 2000. pp. 84–88. DOI: 10.1109/CEC.2000.870279

[4] Shi Y, Eberhart R. A modified particle swarm optimizer. In Proceedings of the IEEE International Conferences on Evolutionary Computation. 1998. pp. 69–73. DOI: 10.1109/ICEC.1998.699146

[5] Yang X-S. Chapter 8 - Particle swarm optimization. Nature-Inspired Optimiza tion Algorithms. 2021:111-121. DOI: 10.1016/B978-0-12-821986-7.00015-9

[6] Samala RK. Analysis of Artificial Intelligence Based Hybridization for Optimal Allocation of Distributed Generations and Economic Power Dispatch in Radial and Mesh Configuration Systems. Iterative International Publishers. ISBN: 978-93-92591-70-9

Chapter 2

Bio-inspired Optimization: Algorithm, Analysis and Scope of Application

G. Devika and Asha Gowda Karegowda

Abstract

In the last few years, bio-inspired optimization techniques have been widely adopted in fields such as computer science, mathematics, and biology in order to optimize solutions. Bio inspired optimization problems are usually nonlinear and restricted to multiple nonlinear constraints to tackle the problems of the traditional optimization algorithms, the recent trends tend to apply bio-inspired optimization algorithms which represent a promising approach for solving complex optimization problems. This work comprises state-of-art of ten recent bio-inspired algorithms, gap analysis, and its applications namely; Particle swarm optimization (PSO), Genetic Bee Colony (GBC) Algorithm, Fish Swarm Algorithm (FSA), Cat Swarm Optimization (CSO), Whale Optimization Algorithm (WOA), Artificial Algae Algorithm (AAA), Elephant Search Algorithm (ESA), Cuckoo Search Optimization Algorithm (CSOA), Moth flame optimization (MFO), and Grey Wolf Optimization (GWO) algorithm. The previous related works collected from Scopus databases are presented. Also, we explore some key issues in optimization and some applications for further research. We also analyze in-depth discussions on the essence of these algorithms and their connections to self-organization and their applications in different areas of research are presented. As a result, the proposed analysis of these algorithms leads to some key problems that have to be addressed in the future.

Keywords: particle swarm optimization, genetic bee colony algorithms, Fish swarm algorithm, artificial algae algorithm, Chicken swarm optimization, Grey wolf algorithm, Cat swarm optimization

1. Introduction

Bio-inspired algorithms nowadays resolve application problems in decision-making, information handling, and optimization purposes from different domains of science and engineering. Many techniques developed fields expected to next few years intelligent optimization algorithms more effective in solving different problems for anomaly and failure detection areas [1]. Optimization plays a major role in more single or multi-objective problems with deterministic or stochastic algorithms [2]. The focus of NP-hard problem-based deterministic or stochastic algorithms to intensification and diversification of meta-heuristic optimization algorithm. Compared to conventional methods, bio-inspired algorithms are intelligent, improved, easy to test, and flexible [3].

In computer networks, security, mechanical problems, electronics image processing, electrical, robotics, production engineering, management, planetary and others are applying bio-inspired algorithms in new era to solve problems easily [4, 5]. Hence it is an emerging field, authors aim to review the discussion and future scope of bio-inspired algorithms. Bio-inspired algorithms concern definitions, principles models, processing steps, merits and demerits reviewed for the most frequently applied bio-inspired algorithms in this chapter. The study discusses bio-inspired algorithms which are purely inspired by identifiable or special behaviour of biological organisms. This chapter covers both emerging and well-known techniques. Ten bio-inspired algorithms: Particle swarm optimization (PSO), Genetic Bee Colony (GBC) Algorithm, Fish Swarm Algorithm (FSA), Cat Swarm Optimization (CSO), Whale Optimization Algorithm (WOA), Artificial Algae Algorithm (AAA), Elephant Search Algorithm (ESA), cuckoo Search Optimization Algorithm (CSOA), Moth flame optimization (MFO), and Grey Wolf Optimization (GWO) algorithm are analysed deeply in this work along with their future scope. Authors have restricted to ten potential algorithms few more potential bio-inspired algorithms is dealt in detail for authors other publications [6, 7]. The work carried on in two phases, in initial phase aims in recognizing algorithms and second phase in depth study of identified algorithms is performed. The chapter noticeably aid in identification of significant bio-inspired solutions for various problems. In section 1, overview of optimization technique and types are presented. Section 2 covers core part of authors work which gives in-depth information on ten bio-inspired algorithms. Section 3 focus on current observation of algorithms and in next section further scope and conclusion are briefed.

2. Overview of optimization

Optimization methods execute and compare iteratively to find solutions for the optimum solutions to be searched. Optimization is part of all problems in all fields. Common types of optimization methods adopted to find solutions are briefed.

2.1 Stochastic optimization

Stochastic optimization (SO) computation involves more vagueness and impreciseness because of randomness in function of minimization or maximization to lend for real-life scenarios. The involved unpredictability exists in form of noise in process of search by Monte Carlo randomness [8]. Stochastic annealing, approximation, programming, swarm-based algorithms are common involved techniques of SO. They include high non-linearity system noise and dimensional models. These models are present to analyze, solve, derive, and numerical extraction of information in resolving decision-making problems. Major investment of SO is in specific applications oriented towards long and short programs. Aircrafts, missile, drug design, and network traffic control applications are getting advantage of SO. Stochastic application tool can be applied as a powerful modelling tool in a few applications, but estimation of real-life problems is another major uncertainity where solving through SO involves practical limitations. Another problem of SO is complete dependency on data available and modelling of it [3, 9].

2.2 Robust optimization

The optimization model is robust based to deal with data to regulate uncertainty. Key features are deterministic, easy computational tractability and set based. Model includes global or local or non- probabilistic or probabilistic models. Any given problem will get involve all the features of robust optimization in order to search for a solution. The technique is also known as the min-max or worst-case approach. Provide a guarantee for solutions to problem application which involves more uncertainty in data. The parameters involved in process of estimation are to resolve estimation errors. One improved model for definition and interpretation is setting more robust constraints [10]. Engineering optimization design results mainly in reliability optimization and feasible input possible values to robust solution structure. Robust optimization gives same weight and values for parametric values in collection of uncertain data. Problems will be resolved with the formulation of cost savings and stability, qualitative and quantitative. Complex problems considered for optimization may extend complexity to a more significant level [6, 7].

2.3 Dynamic optimization

Dynamic programming is another name for dynamic optimization which processes optimal profile of more than one parameter of a system used to find possible solutions for a problem given. Variations of dynamic optimization with optimization discrete time, calculus variation and extended static optimization. The implementation includes a system controller to perform criteria with algorithm to execute control. Dynamic optimization involves a system controller performing optimal substructure and overlapping sub-problems [8]. Dynamic optimization characterizes structure, recursively defines value, computes value and constructs an optimal solution for computation. Dynamic programming optimizes problems and recursively divide problem into sub-problems which can solve either bottom-up or top-down approach. The logic used is general and supple. It solves computation time and storage space [9]. Classification optimization based on different factors is summarized in **Table 1**.

3. Bio inspired optimization algorithms

This section is brief on bio-inspired algorithms detailed. Concept advantage algorithm, flowchart and applications are briefed.

3.1 Particle swarm optimization (PSO)

In proposed particle swarm optimization (PSO) algorithm inspired by intelligent behaviour of birds [11], Craig Reynolds simulated flock social birds behaviour for the first time and later studied by Frank Heppner [12]. PSO search for optimal solution similar to flyting birds with specific velocities determined from previous results and

	Optimization	Factor	Taxonomy
1	Stochastic	Constraints	Unconstrained Constrained
		Nature of equation	Non-Linear Polynomial Linear Quadratic
2	Robust	Physical structure	Optimal control Non-optimal control
		Decision variable permissible value	Integer programming Real-valued programming
3	Dynamic	Variable type	Deterministic Non-deterministic
		Function splitting	Separable Non-separable
		Objective function	Single objective Multi-objective

Table 1.
Classification of optimization.

neighbours in identified search areas [13]. Given a problem identified in search space represent solution in different n-dimension as result in PSO as n particles. The particle moves in n dimension solution space with different velocities. Particles move and store previous behaviours of it and share experiences to store search space. The key merit of POS is its experience to share particle communicate to part or complete swarm to lead motion to detect search space [14]. Each particle will compare the current fitness value with previous optimized results and neighbours in every iteration. Entire particles' global and local algorithm is considered. Each particle best global value is stored as a local value. The entire search space particles' best result is stored as the global best optimal solution. In further iterations value will be adjusted to best optimal if current is best when compared to previous results.

3.1.1 PSO concept

Each particle is running in PSO to identify a feasible solution to the optimization problem in a given search space. The behaviour of flight of particles is considered as search of individual particle. Velocity of particles is dynamically updated based on position of particle and optimal swarm population. The swarm population is composed of M particles in D dimensional space and historical optimal position of the ith particle is represented by pi, iϵ {1,2,3 … M} and optimal position of swarm population is denoted by p_g. In every step velocity and position of each particle are updated dynamically tracking its corresponding previous positions and optimal position of swarm population. The detailed equations are expressed as follows,

$$V_{i_j}^{t+1} = wV_{i_j}^{t+1} + c_1 r_1^t \left(pbest_{i_j} - X_{i_j}^t\right) + c_2 r_2^t \left(gbest_j - X_{i_j}^t\right) \tag{1}$$

$$X_{i_j}^{t+1} = X_{i_j}^t + V_{i_j}^{t+1} \tag{2}$$

In Eqs. (1) and (2) t indicates iteration number , d ϵ {1, 2, 3, … .. D} indicates dimension, $xi,d(t)$ is the dth dimension variable of the ith particle in the tth iteration, and variables $vi,d(t)$, $pg,d(t)$, and $pi,d(t)$ have the similar meanings in turn. ? is inertial weight, and $c1$ and $c2$ denote acceleration coefficients, $r1$ and $r2$ are random numbers uniformly distributed in interval [0, 1]. The objective function to be set, and resultant objective values of each particle correspond to fitness values. These fitness values are used to measure position of particles, historical optimal position of particles and the optimal position of swarm population.

The main concept of PSO is clear from the particle velocity equation that a constant balance between three distinct forces hauling on each particle: (i) particles previous velocity (inertia), (ii) Distance from the individual particles' best-known position (cognitive force) and (iii) Distance from the swarms best known position (social force). These forces are dependent on c1 and c2 weight constants and randomly concerned by r1 and r2 constants. Three forces are shown in vector form as in **Figure 1a** where weight values are specified in vector magnitude. The particles will continue to explore as in search space similar to bird as shown in **Figure 1b** to converge to best position.

PSO shows sufficient better performance on optimization related problems of small scale. The original POS later on improved versions of PSO have been proposed by many researchers. Few incremental works of PSO has been discussed in this sub section which support for large scale and multiple optima [14].

Opposition based PSO discussed by Jabeen et al. [11]. Particle has been classified into two class bad and good. Population of two class generated with fitness computation then original PSO applied. The opposite particle computed using equation

$$Popi = a + b - pi \tag{3}$$

Where in Eq. (3) D is the dimension and R is real number. Quasi-oppositional comprehensive learning particle swarm optimizers (QCLPSO) proposed Chang et al. [7]. Swarm initialization applied by qausi opposite number. The constriction factor balance incremental approach to proposed by Clerc [8]. The equation with constriction factor velocity updated equation is summarized in Eqs. (4)–(6),

$$v_{ij}(t+1) = x\left[v_{ij}(t+1)v_{ij}(t) + \varphi\left(v_{ij}(t)\right) - x_{ij}(t) + \varphi\left(y_{ij}(t)\right) - x_{ij}(t)\right] \tag{4}$$

$$x = 2/|\,4 - \varphi - \sqrt{2\varphi - 4\varphi}| \tag{5}$$

$$\varphi = c1 + c2, \varphi 1 = c1r1 \tag{6}$$

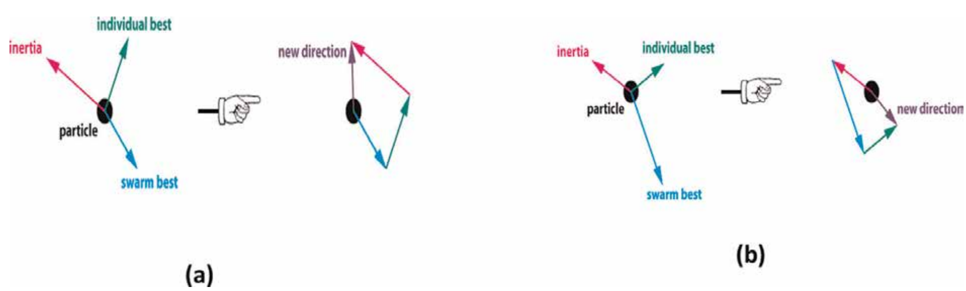


Figure 1.
(a) Exploration of PSO (b) search of new position [4].

```
Initialize the population randomly
 While (Population Size)
{
Calculate fitness If fitness value is better from the best fitness value (pbest) in history then
Update pbest with the new pbest
Select the particle with the best fitness value from all particles as gbest
While maximum iterations or minimum error criteria is not attained
{
 For each particle Calculate particle velocity by equation (1)
Update particle position according to equation (2)
}
 }
```

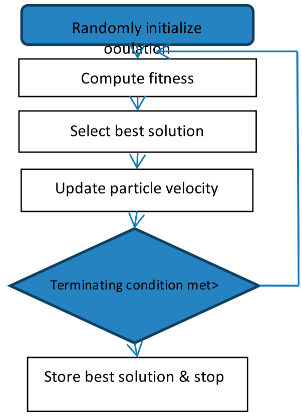


Figure 2.
PSO algorithm & flowchart.

A random value is distributed between [0, 1] for particles by Zhang et al. [13]. The dependency inertia weight to maximize number of iterations to another one is applied to avoid problems in original PSO to search local ability to end. Speed- and accumulation-based inertia weight computation is proposed in Wei et al. [15]. A Cauchy mutation an improved PSO proposed by Wang et al. [14]. The original fitness of particle is selected to mutate the particle to distribute with increase in parameter scale t = 1. It is defined between test function to choose randomly to assign velocity and pop size for particles in swarm. A variation in computation of power distribution among particles applies global best value with power mutation function. The fitness calculation for both particles select appropriate one in Wu et al. [13]. Power mutation function based another opposition-based power mutation function applied for PSO by Imran et al. [15]. Two times mutation being applied on opposite swarms and global best particle power mutation. Global selection for best mutation avoids stagnation. Still improved PSO presented by Imran et al. [16] with student T mutation. Global best particle T student particle identified to work over adaptive and Cauchy mutation (**Figure 2**).

3.1.2 Merits of PSO

- Communication capability: particles can communicate efficiently each other as positions of best particle of all previous iterations are stored.
- Faster Convergence: accelerates more towards optimal solution in optimization
- Simplicity: Updation of velocity and position equations are simple to calculate
- Adaptable to environment: have ability to choose best optimal solutions in changing environment.

3.1.3 Demerits of PSO

- PSO fails to resolve problem which lack in storage and not able to may clear distinction between previous and next particle positions.

- Assume all particles are same hence inertial and velocity also remain same
- PSO do not identify multiple optima.
- Convergence is harder with varied inertia weight

3.1.4 Applications

PSO has been applied in most domains to optimize solutions from agriculture to industry. PSO has been extensively applied in different geotechnical engineering aspects such as slope stability analysis, pile and foundation engineering, rock and soil mechanics, and tunnelling and underground space design [17]. PSO has been widely used in various kinds of planning problems, especially in the area of substation locating and sizing [14]. But in the area of heating supply, PSO is mainly applied in heating load forecasting [18–20], but rarely used in Heat System Planning. PSO can be applied for various optimization problems, for example, Energy-Storage Optimization. PSO can simulate the movement of a particle swarm and can be applied in visual effects like those special effects in Hollywood film.

3.2 Genetic bee colony (GBC) algorithm

Bee food identification and collection intelligent swarm technique is defined in artificial bee colony. The best bee for the required problem is selected through parameters communication link, task allocation, reproduction, dance, placement mating and movement. GBC is optimised towards solution iteratively in an attempt to increase efficiency for any critical problem. Bee swarm is categorized as employee, onlooker and scouts. The employee bee identifies fresh sources of food. Scout bees job is to assign fitness quotient to entrust job of random search for employee bee identified spots. The assignment is random-based. If freshly identified food is better than earlier findings then, bees will collect from fresh location. Constantly employee bees look for best site for food collection. The onlooker bee is responsible to identify the best food source considering quantitative factor of food availability [21, 22].

3.2.1 Concept

The ABC algorithm consists of four main steps: initialization, employed bee phase, onlooker bee phase, and scout bee phase. After the initialization step, the other three main steps of the algorithm are carried out repeatedly in a loop until the termination condition is met. The main steps of the ABC algorithm are as follows.

Step 1 (initialization). In the initialization step, the ABC generates a randomly distributed population of *SN* solutions (food sources), where *SN* also denotes the number of employed or onlooker bees. Let φ represent the ith food source, where is the problem size. Each food source is generated within the limited range of ith index by where$\varphi = 1,2, \ldots SN, j = 1,2, \ldots .D, \varphi i, j$, is a uniformly distributed random real number in , x_{min} and x_{max} and are the lower and upper bounds for the dimension , respectively. Moreover, a trial counter for each food source is initialized as in Eq. (7).

$$x_{i,j} = x_j^{min} + \varphi_{i,j}\left(x_j^{max} - x_j^{min}\right) \tag{7}$$

Step 2 (employed bee phase). In the employed bee phase, each employed bee visits a food source and generates a neighboring food source in the vicinity of the selected food source. Employed bees search a new solution, by performing a local search around each food source as follows: where is a randomly selected index and is a randomly chosen food source that is not equal to; that is, is a random number within the range generated specifically for each and combination. A greedy selection is applied between and by selecting the better one as in Eq. (8).

$$v_{i,j} = x_{i,j} + \varnothing\left(x_{i,j} - x_{r1,j}\right. \tag{8}$$

Step 3 (onlooker bee phase). Unlike the employed bees, onlooker bees select a food source depending on the probability value, which is determined by nectar amount associated with that food source. The value is calculated for the food source as follows considering Eqs. (9) and (10): where the fitness value of solution and calculated as in (4) for minimization problems. Different fitness functions are employed for maximization problems. By using this type of roulette wheel based probabilistic selection, better food sources will more likely be visited by onlooker bees. Therefore, onlooker bees try to find new candidate food sources around good solutions. Once the onlooker bee chooses the food source, it generates a new solution using (2). Similar to the employed bee phase, a greedy selection is carried out between.

$$p_i = \frac{fit_i}{\sum_{j=1}^{SN} fit_i} \tag{9}$$

$$fit_i = \begin{cases} \dfrac{i}{1 + fit_i}\, fi \geq 0 \\ 1 + abs(fi)\, fi < 0 \end{cases} \tag{10}$$

Step 4 (scout bee phase). A trial counter is associated with each food source, which depicts the number of tries that the food source cannot be improved. If a food source cannot be improved for a predetermined number of tries (limit) during the onlooker and employed bee phases, then the employed bee associated with that food source becomes a scout bee. Then, the scout bee finds a new food source using (1). By implementing the scout bee phase, the ABC algorithm easily escapes from minimums and improves its diversification performance.

It should be noted that, in the employed bee phase, a local search is applied to each food source, whereas in the onlooker bee phase better food sources will more likely be updated. Therefore, in ABC algorithm, the employed bee phase is responsible for diversification whereas the onlooker bee phase is responsible of intensification. The flow chart of the ABC is given in **Figure 3**.

3.2.2 Merits of GBC

The ABC algorithm is a population-based algorithm with the advantages of finding global optimization solution, being simple and flexible, and using very few control parameters. The ABC algorithm has been applied to many real-world applications, for example, function optimization, real-parameter optimization, digital filter design, clustering, and neural network training. ABC algorithm-based applications are easy to

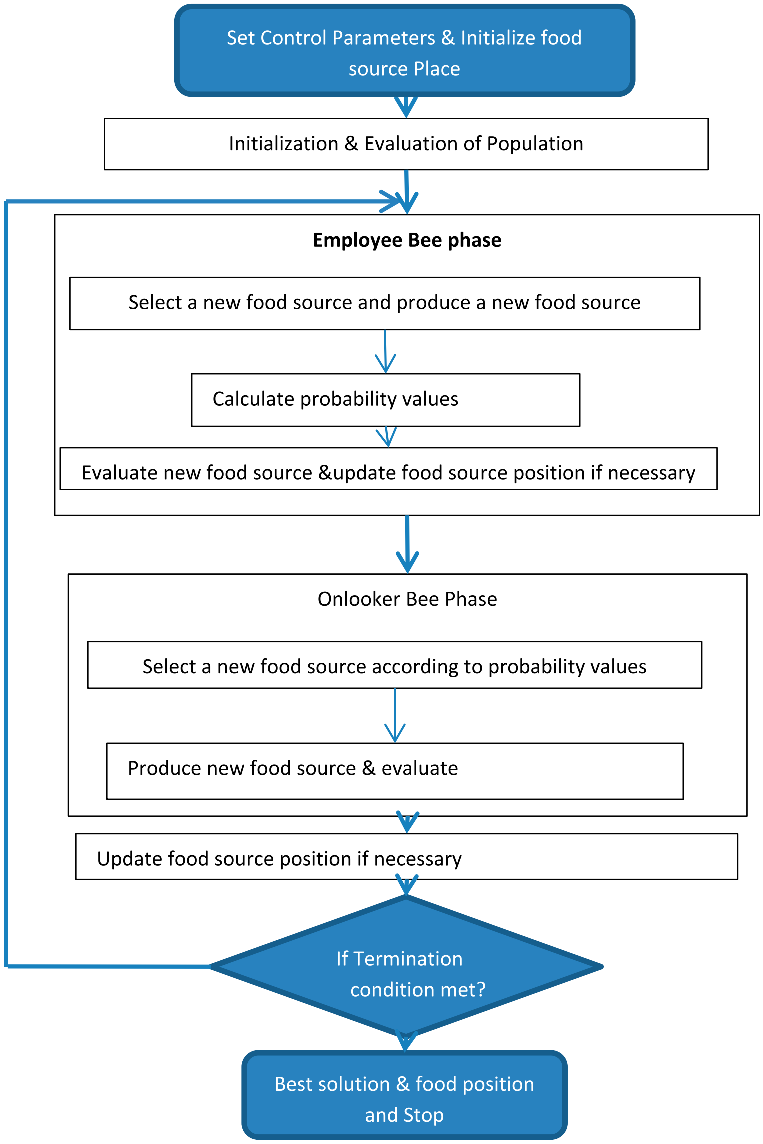


Figure 3.
GBC flowchart.

build, robust, converge fast, flexible and time efficient. Compared to PSO ACO parameters considered in ABC is less.

3.2.3 Demerits of GBC

Inspite of mentioned advantage GBC has few problems also. GBC are slow convergent speed in large computations and accuracy is less. GBC may face premature convergence problem for more duration application. The population size is fixed, and size is variations or non-autonomous. Individuals can extend the searching space and increase the probability of finding global optimization solution; however, it costs much time in each generation; oppositely, it may obtain a local minimum.

3.2.4 Applications of artificial bee colony (ABC) algorithm

GBC are more affine towards single-objective numerical optimization problems but not limited and can be extended to multi-parametric. Decision making, time schedule, assignment, search, inception, boundary setting, and network issued are other more applied fields of GBC [21]. GBC is capable to handle constrained and unconstrained, continuous and discrete, differential and non-differential oriented problems [22]. GBC is not specific in domain, applicable from agriculture to industry, and rural area to military field.

3.3 Fish swarm algorithm (FSA)

Fish swarm algorithm inspired by the behaviour of movement of aquatic fish in liquid medium. The target is picked randomly and moves toward in an iterative manner. Visually shorter distance considered an initial step, influences on final step. Initial values remain constant and considered along parameters. Suitable initial value selection leads toward the best optimum solution. Fishes are capable of venturing into bigger steps in search of a larger environment where they exist. So, fish is capable of escaping from unfavourable circumstances at any stage. But some deficiencies in large values may cause low steadiness. Global search is potential factor in generating local search with a larger visual position of FSA. Better fitness can be found for better fitness to search for parameters to make the algorithm steady and accurate. Fish are capable of moving quickly towards the target and can get passed from local best search results. FSA algorithm design has undergone many changes in design in order to fulfil the needs of different types of problems. The variation in algorithm can be grouped into solutions of FSA for continuous and discrete, combinatorial and binary, multi-parametric and hybrid FSA. Fei et al. [23] selected nine search positions to initialize the Afs for motion estimation. Zhu et al. [24] and Gao et al. [25] used the chaotic transformation [26] method to generate a more stable and uniform population. Kang et al. [27] used a uniform initialization method to initialize the population, while Liu et al. [28] initialized the

Afs based on the optimization problem in hand. The MSAFSA [29] model introduced both the leaping and swallow behaviors to escape from the local optima and reduce, Yazdani et al. [30, 31] introduced mNAFSA for optimization in dynamic environments.

3.3.1 Concept

Fish Swarm algorithm and background is discussed. Notations used are X, V, S, X_v indicate current position of fish, distance, step, position, respectively. N visual fishes are indicated as X1, X2, X3 … … .. Xn. Y = f(X) denotes the food concentration of the AF at the current position, di;j =||Xi – Xj ||. The FSM involves four key operations: preying behaviour, swarming behaviour, following behaviour and random behaviour. Preying is fish behaviour to move itself towards high concentration of food. It is represented mathematically as in Eq. (11) considering with in visual distance ith fish. The fish preying will continue to try number of times if not satisfied within, then

DOI: http://dx.doi.org/10.5772/intechopen.106014

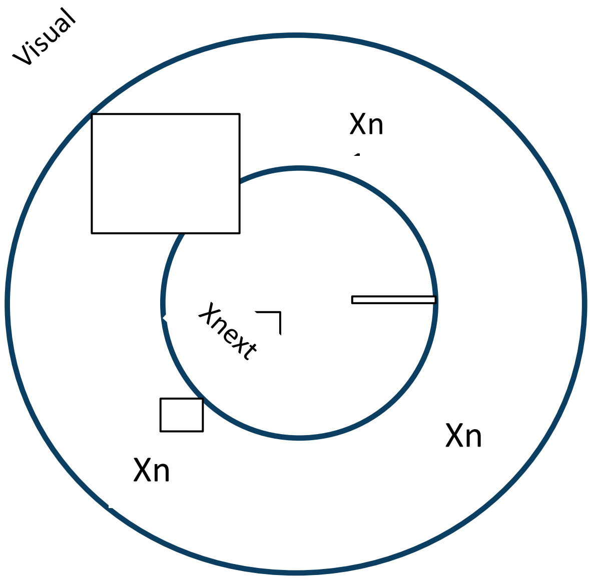


Figure 4.
FSA visual.

randomly computed using Eq. (12). Fishes group among themselves to from any danger situations against them. Mathematically, central position in fish swarm is computed as in Eq. (13). Fish when a locates good concentration of food. The preying movement for fish in step movement is represented in Eq. (14). Few fishes randomly move freely if lie in sparely concentrated food. This behaviour is modeled as in Eq. (15) (**Figure 4**).

$$X_j = X_i^{(t)} + V * rand() \tag{11}$$

$$X_i^{(t+1)} = X_i^{(t)} + V * rand() \tag{12}$$

$$x_{cd} = \frac{\sum_{j=1}^{nf} x_{jd}}{n_f} \tag{13}$$

$$X_i^{(t+1)} = X_i^{(t)} + S * rand() * \left(\frac{X_j - X_i^{(t)}}{\|X_j - X_i^{(t)}|(t)|} \right) \tag{14}$$

$$X_i^{(t+1)} = X_i^{(t)} + V * rand() \tag{15}$$

3.3.2 Algorithm and flowchart

FSA perform record one if new. This search continues until end is not met following four operational steps as mentioned in previous section. The algorithm FSA is shown in briefed in **Figure 5**.

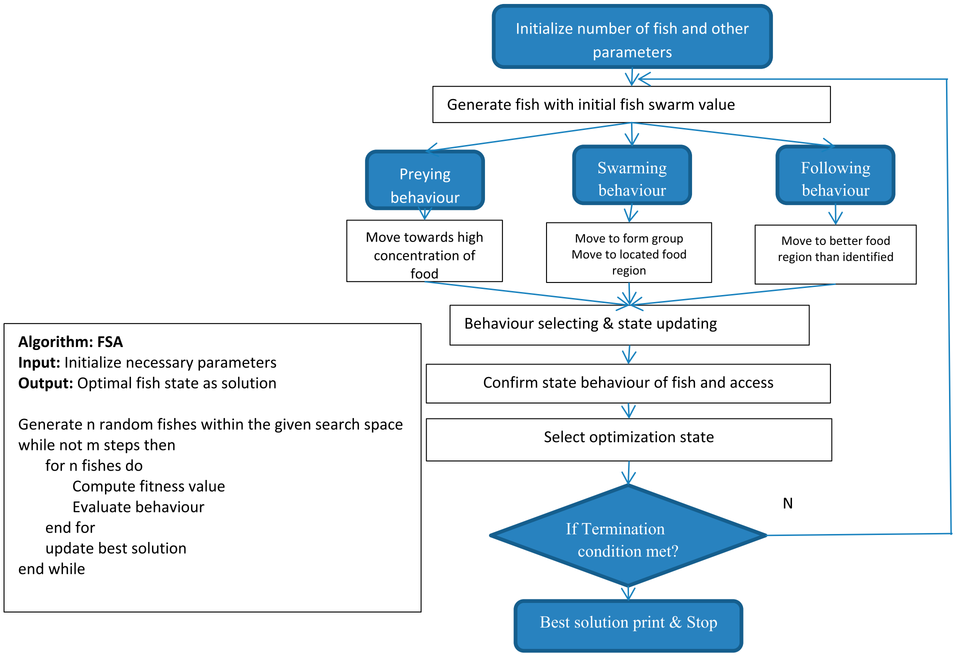


Figure 5.
FSA algorithm & flowchart.

3.3.3 Merits

FSA similar to GBC has got increased convergence power and flexible. In addition, it exhibits fault to tolerant and accuracy feature. Global search ability, tolerance of parameter setting and robustness are other merits of FSA. It solves nonlinear and multi modal problems.

3.3.4 Demerits

FSA exhibits high complexity, lack of balance among one is ineffective if lack balance between local search. Not suitable for global convergence problem. The information transfer if experience low search rate. As good robustness, global search ability, tolerance of parameter setting, and it is also proved to be insensitive to initial values

3.3.5 Applications

FSA has been applied for network related problems, control of resources, image processing related problems. In order to increase evolutionary capabilities of FSA, in few swarm solutions hybrid to FSA.

FSA incorporate to solutions in wireless sensor networks [30, 32, 33], tracking [34], medical estimations [35–37], segmentation [32], clustering [33], regression [38], image processing [39, 40], calibration [35], localization [41], power systems [36, 42].

3.4 Cat swarm optimization (CSO)

Chu et al. introduced cat swarm optimization technique to solve most of engineering problems inspired by the movement of cats. The process is carried on in two different modes seeking and tracing modes. Nodes virtually move in fixed areas as cats to determine optimal solution. Number of virtual cats are fixed in both modes and predefined in few cases ratio known as MR. the N virtual cats is placed randomly. Processed and unprocessed cats are identified for each dimension based on its value of MR set either to 0 or 1 for tracing or seeking in coming rounds. Every cat compute fitness function in evaluation then among the existing best will be chosen initially existing is compared if It best retained for fitness function otherwise coordinates will be changed to new best cat. The movement of cat adjusted towards solution space identified as identified in initially. Choose for unprocessed cats in tracing mode through permutation. Tracing mode ends if no more cats are left. Traced coordinate nodes will be selected as best solution at end. In seeking mode, cats' movement will be slow and conformist. Essential parameters of seeking mode are seeking memory pool, Ra range of identified dimension, counts of dimensions to change and self-position. Improved CSO algorithm proposed by Tsai et al. supports parallel information exchange in tracing mode. The parallelizing of virtual agents is adopted in PCSO [43, 44]. PCSO finds application in parallel processing inspired by colonies of cats tracing for food.

3.4.1 Concept

CSO identification of optimized solution is described in this sub section step by step. The seeking feature of cats carried in five processing steps. In first step, j copies of cat generated is recognized applying Eq. (16). Addition or decrement of SRD value on selected search space defined by Eqs. (17)–(19) in step two. In next step, fitness value for all candidates is selected. In step four, calculation of probability of cat performed by Eq (20). Sort and select best solution by roulette wheel selection in last step. In tracking mode cats imitate movement of prey during tracing. This process can be discrete into three operational steps. In first step, velocity of each cat is updated as in equation 6 for given search space. The random value for cat adjusted in range 0-1. In step 2, the valued are rearranged based on velocities of cat. Velocities are set to maximum velocity value. Position of cat is updated selecting by Eq. (7) in last step.

$$j = \begin{cases} SMP & SPC = \text{"}true\text{"} \\ SMP - 1 & SPC = \text{"}false\text{"} \end{cases} \tag{16}$$

$$M = Modify \cup (1 - Modify) \tag{17}$$

$$|Modify| = CDC * M \tag{18}$$

$$x_{jd} = \begin{cases} x_{jd} & d \nexists Modify \\ (1 + rand * SRD) * x_{jd} & d \in Modify \end{cases} \tag{19}$$

$$p_i = \begin{cases} 1 & when\ FS_{max} = FS_{min} \\ \dfrac{|FS_i - FS_b|}{FS_{max} - FS_{min}}, & whare\ 0 < i < j,\ otherwise \end{cases} \tag{20}$$

$$v_{k,d} = v_{k,d} + r.c.(x_{best,d} - x_{k,d}), d = 1,2\ldots..M \tag{21}$$

$$x_{k,d} = x_{k,d} + v_{k,d} \tag{22}$$

3.4.2 Algorithm and flowchart

The algorithms and flow of operations of CSO is summarized in **Figure 6**.

3.4.3 Merits

COA is Simple to construct and have minimal parameters to adjust. COA has got ability to execute in parallel system. The design is robust. Can converge fast, find global solution, overlap and mutate. Have computational time less. Find accurate mathematical models. Discover good and rapid solutions. Adapt changes in new system and dependent on random decisions.

3.4.4 Demerits

Definition of initial parameters is time consuming. COA does not work better for scattering problems and can converge at faster rate if trapped in complex problems. The time to converge and towards convergence for multi objective and larger sized problem is more.

```
Algorithm  : CSO
Input: Number of cat,
Output: Best cat as solution
Begin
    Determine best MR applying minimal rounds
    Create cat population
    While not termination condition do
        For size n do
            If cat is in seeking mode then
                Apply seeking mode
            Else
                Tracing mode
        End for
    Evaluate best fit
    End while
    Store best solution
end
```

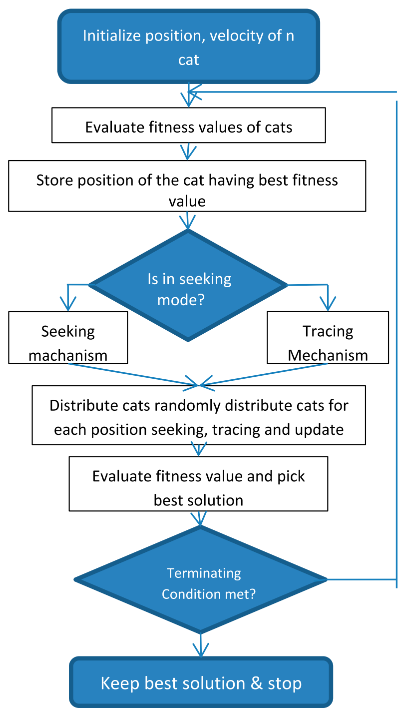


Figure 6.
COA algorithm & flowchart.

3.4.5 Applications

CSO optimization is being incorporated in media for information hiding [44], aircraft scheduling recovery in limited processing time [45]. Voltage stability, economically dispatch in transmission system, hybrid generation systems, task allocation, data mining, project scheduling, optimal contract capacity, global numeric optimization problems. Applied for clustering technique in green expression classification, travelling salesman problem, data hiding, graph coloring, SVM, K means. CAO even find its application in stock market and supply chain in currency exchange rate analysis and stock prediction. COA is applied in image processing for machinery fault detection, plant modeling, image edge enhancement, water marking and single bit map. COA extended application in electronics for cognitive radio engine cooperative, spectrum sensing, linear antenna array synthesis, aircraft maintenance, routing for wireless sensor network.

3.5 Whale optimization algorithm (WOA)

Whale optimization algorithm proposed by Mirjalili et al. WOA is also based on population of whale. It simulates bubble-net attacking method of humpback whales when hunting their preys. Whales are intelligent due to the spindle cells in their brain. They live in group and are able to develop their own dialect. Whale optimization algorithm consists of two modes of operation. The two modes of operation named as exploitation and exploration. In first prey encircling and position update in spiral manner carried on. Searching for prey randomly done in second phase [46–48]. WHO exploitation phase for prey encircling is mathematically equalized as bubble net attack system. Humpback characteristics of whales considered for phase one behaviour. Whales encircle prey with identification of them in an undefined search space. Initial solution of nearby prey or ideal assumed as best further best solution will be updated once exploration begins. Distance between prey and whales calculated initially then, updates for spiral positioned distance to it. WOA has modified and incorporated improvements by many researchers [49–53]. Few notable changes included in AWOA, IWOA, chaotic WOA, ILWOA, and MWOA research work. WOA hybridized with other meta-heuristic algorithms PSO, BA, and others in order to improve local search [34, 54–56].

3.5.1 Concept

Whale has a special hunting mechanism which is called bubble-net feeding method. This foraging behaviour is done by creating special bubbles in a spiral shape or nine shape path. Humpback whales know the location of prey and encircle them. They consider the current best candidate solution is best obtained solution and near the optimal solution. After assigning the best candidate, the other agents try to update their positions towards the best search agent as computed by Eq. (23). In Eqs. (23) and (24), t is the current iteration, A and C are coefficients vectors, X^* is the position vector of the best solution. The vector A and C are calculated using Eqs. (25) and (26). In Eqs. (25) and (26) a are linearly decreased from 2 to 0 over the course of iterations and r is random vector in [0, 1]. The humpback whales attack the prey with the bubble-net mechanism in exploitation phase. In shrinking encircling mechanism, the value of A is a random value in interval [-a, a] and the value of a is decreased from 2 to 0 over the course of iterations. Spiral updating position mechanism calculate the distance between the whale location and the prey location is calculated then the helix-shaped movement of humpback is created using Eq. (28). D' = |X*(t) – X(t)| is distance between the prey and the ith whale, be is a constant, l is random number in [−1, 1]. Whale selectively applies swim around prey techniques suitably. The mathematical model of these two mechanisms assumes to choose between these two mechanisms to update the position of whale as in Eq. (29). In steady exploitation phase the humpback whales search for prey and change their position of whale. The force the search away from reference whale the mathematical model of exploration is computed as in Eqs. (29) and (30).

$$X(t+1) = \begin{cases} X*(t) - A > D \; if < 0.5 \\ D^t.e^{bl}.\cos(2\pi l) + X*(t) \; if \; p \geq 0.5 \end{cases} \tag{23}$$

$$D = |C.X*(t) - X(t)| \tag{24}$$

$$X(t+1) = X*(t) - A.D \tag{25}$$

$$A = 2a.r.a \tag{26}$$

$$C = 2.r \tag{27}$$

$$\mathrm{X}(\mathrm{t}+1) = D^t e^{bl}.\cos(2\pi l) + X*(l) \tag{28}$$

$$D = |c.X_{rand.X}| \tag{29}$$

$$X(t+1) = X_{rand} - A.D \tag{30}$$

3.5.2 Algorithm and flowchart

The detailed workflow and algorithms is presented in **Figure 7**.

```
Algorithm :Whale Swarm Algorithm
Input: Number of whale, position,variable v=0,u =infinity
Output:  Optimal solution
Begin:
     Initialize parameters
     Initialize whale positions
     Compute fitness value of all whale
     while not final criterion do,
          for n whales do,
               Find the better and nearest whale  among all n
               For i in m objective values do
                    If u not equal to m then
                         If f(m_i) < 0.5 then
                              If dist(m_i, m_v) < u then
                                        v= i & u dist(m_i, m_v)
                              end if
                         end if
                    end if
               end for
               if Solution exist then
                    N moves under guidance of best near whale
                    Evaluate fitness value
               end if
          end for
     end while
store better optimal solution
end
```

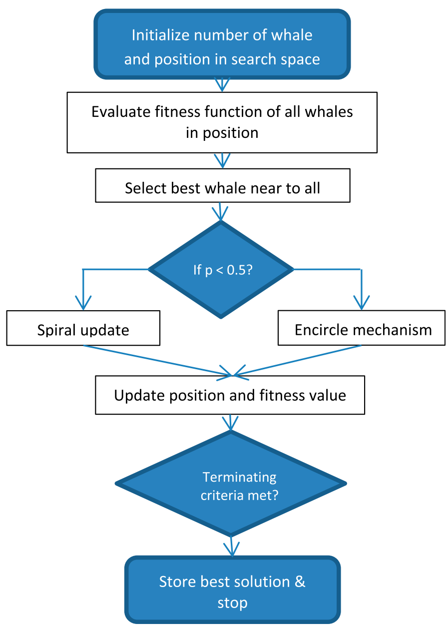


Figure 7.
WOA algorithm & flowchart.

3.5.3 Merits

Whale optimisation avoid problem of local optima have got ability to compute local and global optima for any constrained or unconstrained optimization applications. During the process, it even does not require any structural are parametric rearrangement or alteration in value. The exploration for best solution computed simply and easily at faster rate. Improve quality of generated population and converge at faster rate.

3.5.4 Demerits

WOA is not suitable for larger spaced problem incurs more time to explore and converge. The accuracy of solution is questionable. The optimal solution cannot be recognized for optimization of problems to solve high dimensional problem. Randomization technique of core WOA solution is complex. Balance among process of exploration and exploitation is lacking. The encircle mechanism slows WOA to jump from one local optima to another yielding low performance. Application problems of classification and dimensional reduction problem

3.5.5 Applications

WOA has ability to incorporate in dynamic applications. Most of researchers applied WOA for electrical, mechanical and management problems. WOA has been used to solve problem in engineering, multi objective, binary, identification classification and scheduling. WOA has found problems Power plants and systems scheduling [57] has confirmed to standard radial systems. Verify test system in execution of IEEE 30-bus [58, 59]. Size of pillars and optimization to increase efficiency of building is analysed [60]. Energy rise of solar energy to get importance in design of photovoltaic cells. WOA benefit solar cell and photovoltaic cells [61] by calculating internal parameters automatically. The partially cloudy atmosphere traced to get highest power region by a modified artificial killer whale optimization algorithm MAKWO [62]. Medical image analysis for classification and diagnose liver and cluster based abdominal to avoid intensity values to overlap [63]. WOA incorporated in economic and emission dispatch [64], vehicle fuel consumption [65], mobile robot path planning [66], optimal allocation of an ameliorative of water resource [67], design problem [68], heat and power economic dispatch [69].

3.6 Artificial Algae Algorithm (AAA)

Artificial Algae Algorithm initially proposed in 2015 by Uymaz et al. is also a meta-heuristic bio inspired algorithm. Microalgae growth and reproduction in presence of sunlight behaviour are considered in algorithm AAA. Algae swim towards presence of sunlight for food production following process as photosynthesis. The movement of algae towards sunlight will be in helical manner. They live in groups as algae colonies. The algae identify best sunlight presence to carry on photosynthesis itself considering largest size and reproduce algae's with highest energy. In case sunlight presence is less, then size of algal colony and energy level is less and starts for high starvation level. If sunlight is less algae colony tries to adopt itself in environment for its survival otherwise algae cells die because of starvation. The adaptation of algae cells in unsupportive environment is known as evolution [70]. Uymaz et al. developed AAA then they modified to perform better [71]. From then many researchers contributed for AAA by incorporating AAA in different fields. Multi-objective optimization for AAA designed by Babalık et al. [72]. Binary version presented by Zhang et al. [73]. AAA applied in various fields from processing to manufacturing and in applications ranging from agriculture to home [74]. Few researcher improved AAA through hybridization [75].

3.6.1 Concept

AAA proposed for first by Uymaz et al. deals with considering advantages of research area of the properties found in algae. Algae moves from helically towards lighter sources. Algae adopt in nature to adapt and reproduce forming colonies which represent a solution. Colony of algae consists set of cells which dwell together. The colony exposed to external forces. The algae are divided into group and each become new colony as can move jointly, under in appropriate circumstance to from new colony. AAA process incorporated by three parts: Evolutionary, adaptation and helical movement. In evolutionary process, algae colony grows and flourish to get sufficient light, and benefit conditions. The algae undergo mitosis to result in two new algae. If not algae will perish under less nutrition and lighter conditions. In few scenarios if algae cannot grow in an environment due to lack of supporting factors. In such environments algae adapt by itself to environment in order to survive as other species. Finally, algae if it could not adapt then moves toward large grouped algae. If starvation occurs algae stop to adopt. Algaes move in helical movement by swim. In order to live they try to remain close to surface of water to get light. The search capacity will not remain same. Algae growth is more in region where frictional surface is more. The chance of algae movement is more in fluid. Helical motion supports to move algae at higher rate. The energy in different surfaces is not constant and is directly proportional to quantity of food and type of nutrient available in the environment. Capability and survival of algae existence depend on its adaptation and movement. The algae survival process mathematically applied in functional parts. Initially fix size of algae by Eq. (31). Evaluate fitness value of algae and size of colony by Eq. (32). Adaptation of algae is through growth of algae and use of nutrients by Eq. (33). The energy of algae computation inclusion of frictional force is computed applying Eqs. (33) and (34). During adaptation process algae build itself under non favorable or movement to nearby stronger and larger algae colony part. The optimization for given problem can be computed by Eqs. (35)–(38). The three subgroups of algae considered for adaptation. Identification of starvation be Eqs. (39) and (40). Section of best solution is selected by Eq. (41).

$$X_{ij} = LB_j + \left(UB_j LB_j\right).RAND\ i = 1,2,3\ldots..N; j = 1,2,3\ldots.D \tag{31}$$

$$\mu_i = \frac{S}{K_s + S} \tag{32}$$

$$G_i^{t+1} = \mu_i^t G_i^t\ \mathrm{i} = 1,2,3,\ldots\ \ldots.\mathrm{N} \tag{33}$$

$$\tau(x_i) = 2\pi \sqrt[3]{\frac{3G_i}{4\pi}}^2 \tag{34}$$

$$GE^{t+1} = norm\left(\left(rank\left(G^t\right)\right)^2\right) \tag{35}$$

$$X_{i_m}^{t+1} = X_{i_m}^t + \left(X_{j_k}^t - X_{i_k}^t\right)\left(\Delta - \tau^t(X_i)\right)P \tag{36}$$

$$X_{i_k}^{t+1} = X_{i_k}^t + \left(X_{j_k}^t - X_{i_k}^t\right)\left(\Delta - \tau^t(X_i)\right)\cos\alpha \tag{37}$$

$$X_{i_l}^{t+1} = X_{i_l}^t + \left(X_{j_l}^t - X_{i_l}^t\right)\left(\Delta - \tau^t(X_i)\right)\sin\beta \tag{38}$$

$$Starving^t = maxA_i^t\ i = 1,2,3, \dots \dots .N \tag{39}$$

$$Starving^{t+1} = Starving^t + \left(Biggest^t - Starving^t\right).rand \tag{40}$$

$$biggest^t = maxA_i^t\ i = 1,2,3 \dots \dots \dots .N \tag{41}$$

3.6.2 Algorithm and flowchart

The algorithm and flow of operation of AAA is shown in **Figure 8**.

3.6.3 Merits

AAA exhibits accuracy for identified colonies. Converge faster towards local and global solution compared to ACO or PSO. Algorithm is convenient and efficient. The method helps find efficient and high accurate result. Produce robust algorithm for real-time optimization problems. Main benefit for gradient-based problems provide by an efficient optimize in few steps and simple to generate.

```
Algorithm : Artificial Algae Algorithm
Input: number of algae and colony
Output: Best solution and colony

Begin
Generate an initial population of n algal colonies with random solution
evaluate f(xi), i = 1, 2, …, D
while stopping condition not reached do
for n algae do
while energy of ith colony not finished do
modify the colony with helical movement
end while
end for
apply evolutionary
apply adaptation
apply helical movement
end while
keep best solution
end
```

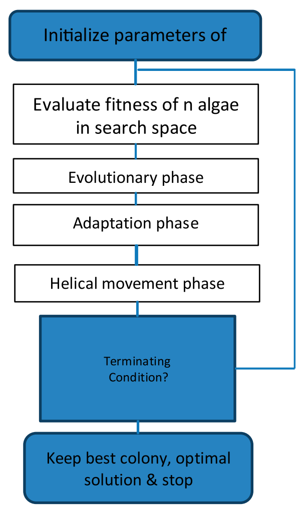


Figure 8.
AAA algorithm & flowchart.

3.6.4 Demerits

Major problem of AAA is its expensive apparatus, consumption of time and specialized operator. If data and input size increases accuracy will be minimized. They tend to stick to local optima, increased dependency. AAA apply randomness by this methodology is simple but result accuracy is questionable. Hence applications involved in AAA are complex and provide unstable result.

3.6.5 Applications

In optimal placement distributed power flow controller (DPFC) with MCFC, optimal coverage, routing and selection of cluster head in wireless sensor network.

3.7 Elephant search algorithm (ESA)

Elephant search algorithm developed by Adams et al. is inspired by elephant search for water. Normally elephants search for water in drought within swarm. Elephant swarm together search water source. Each elephant swarm consists of leader responsible to make decision regarding movement of whole group. Elephant is identified by its particular position and velocity in each group very similar to other swarm techniques. Leader elephant informs rest of elephants in group in case best water source is identified. The communication is through chemical, tactile, acoustic or visual means. The fitness function is computed considering water source quality and quantity. The elephants' group can move from one water source to another and visits previous also if necessary as they got good memory. Group visit previous water source in case older identified is best solution in compassion to new water source. Elephants search for best solution locally and globally then best solution will be identified in given solution space following long and short distance communication. Switching probability is key controller in considering water search either local or global.

3.7.1 Concept

EHO is meta-heuristic simulated behaviour in herds of elephants [24] introduced by Wang. Optimize solution for global optimization tasks [5]. Each solution I in each clan ci is updated considering current information such as position and matriarch. The generations are updated by algorithm execution through separating operators. Each individual in heard represent vectors in 2D. The dimensions in unknown population are included. The population is divided into n clans. Updating operator is modeled by increment or decrement each solution i in the clan by ci by influence of ci to identify best fitness value in generation. Fitness update solution in each clan c_i represented in Eq. (42). New and old position in clan, incremental factor based on influence of matriarch are parameters included of Eq. (43). In 2D the central clan is computed through Eq. (44). It updates individual value of elephants in heard. The total search space indicates number of solutions in clan. The separating search space and n_{ci} indicates number of solutions in clan in c_i. The separate operator is applied at each generation for execution on worst individual in population. Choose random population [0-1] be uniform distribution range within lower and upper limits of the position of the individual by Eq. (45).

$$x_{new,c_i,j} = x_{ci,j} + \alpha.\left(x_{best,c_i} - x_{c_i,j}\right).r \tag{42}$$

$$x_{new,ci,j} = \beta.x_{center,c_i} \tag{43}$$

$$x_{center,ci,d} = \frac{1}{n_{ci}} . \sum_{j=1}^{d} x_{c_ij,d} \tag{44}$$

$$x_{worst,c_i} = x_{min} + x_{max} - x_{min} + 1) * rand \tag{45}$$

3.7.2 Algorithm and flowchart

The detailed algorithms and flow of operation of EHOA is presented in **Figures 9** and **10**.

3.7.3 Merits

EHOA is more performance stable than other meta-heuristic algorithms such as PSO. Convergence is faster because they are in herd. Have ability to search a population in parallel. Rapidly discover good solutions similarly adapt to changes such as distance. The computation is simple. EHOA is efficient in solving problem which are difficult to find accurate mathematical models. Computational time is less and overlap is avoided.

3.7.4 Demerits

Probability can change for each iteration, theoretical analysis is difficult, and sequence of random decisions are major hindering factors of EHOA. Time requirement for convergence is uncertain.

```
Algorithm 1 Pseudo-code of EHO algorithm
Input: n elephant and max. generation
Output: Best solution
Begin
Generate population and calculate fitness
 Divide population into m clans
Calculate fitness of each individual
For n individuals
While not termination condition do
Sort all solution according to their fitness
For all clans do
Update generation and find fitness
End for
Update best
End while
End for
Select best if existing is best retain
end
```

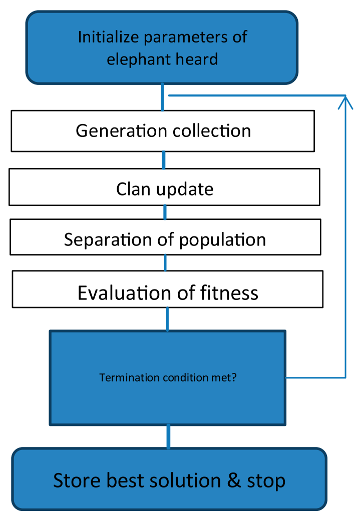


Figure 9.
EHOA algorithm & flowchart.

Algorithm : Cuckoo search algorithm
Input: n nests with eggs
Output: best egg as solution
Begin
Objective function f(X), X = (x1, ..., xd)^T
Generate initial population of n host nests X_i (i = 1, 2, ..., n)
while (t <MaxGeneration) or (stop criterion)
Get a cuckoo randomly by L´evy flights
evaluate its quality/fitness F_i
Choose a nest among n (say, j) randomly
if (F_i> F_j),
replace j by the new solution;
end if
A fraction (p_a) of worse nests are abandoned and new ones are built;
Keep the best solutions
Rank the solutions and find the current best
end while
Postprocess results and visualization
end

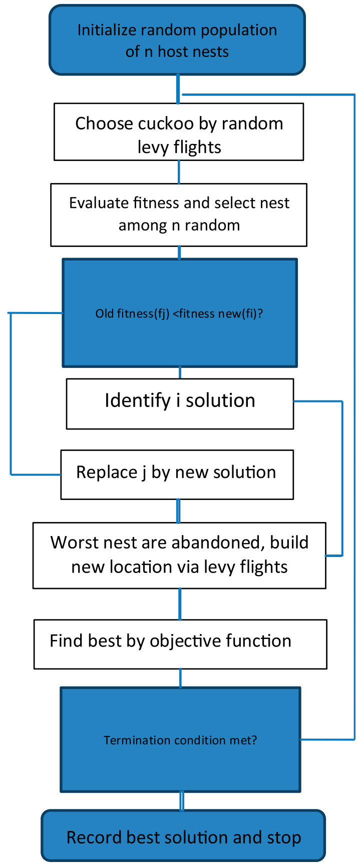


Figure 10.
CSO algorithm & flowchart.

3.7.5 Applications

EHO applied to optimize training artificial neural network, selection structure and weight for neural networks, training neural netwosk, optimizing underwater sensor networks, unmanned aerial vehicle path planning, clustering, support vector machine, control problem.

3.8 Cuckoo search optimization algorithm (CSOA)

Yang and Deb introduced cuckoo optimization in 2009 a meta-heuristic algorithm. Later Gandomi, Yang, and Alavi and Yang and Deb extended to solve single or multi-objective problems involved in any constraints or complexity. The solution is capable to resolve potential solutions of any randomly selected population in habitants of cuckoo. The function of CSOA is global optimality, real-world problems are NP-hard for problem used in any problem. Construct workable solution required to be globally optimal solution replicating behaviour of cuckoos. They lay eggs in nest of other birds and obliterate eggs of birds to guarantee hatching of its breed. Cuckoos brood parasitism is simulated in three different ways: Intra-specific brood parasitism, nest take over and co-operative breed. The basic cuckoo search algorithm has undergone changes convergence speed of cuckoo search algorithm is increased in modified cuckoo search [74] by avoiding cross overs. Binary version of cuckoo search algorithm is presented in [75] to increase accuracy by reducing problem to binary coordinated feature. In [76] improves cuckoo search by resetting position and random vector value of eggs rather considering as static parameter value.

3.8.1 Concept

CS algorithm is based on the obligate brood parasitic behaviour of some cuckoo species in combination with the levy flight behaviour of some birds and fruit flies. Some species of Cuckoo birds lay their eggs in communal nests. If a host bird discovers the eggs are not their own, they will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. CS, can be described using following three idealized rules:

a. Each cuckoo lays one egg at a time, and dump its egg in randomly chosen nest;

b. The best nests with high quality of eggs will carry over to the next generations;

c. The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host birth a probability pa $\in [0, 1]$.

3.8.2 Algorithm and flowchart

The algorithm and flow of operations of CSOA is presented in **Figure 10**.

3.8.3 Merits

A meta-heuristic method exhibits several advantages as easier for applications to change parameters to meet requirement of applications. It is very easy fall for optima of local solution to slow convergence rate. In addition, cuckoo search is simple and easy to follow with real-world engineering applications. Cuckoo search algorithm easy to implement in comparison to other population algorithms.

3.8.4 Demerit

CSA is about easy to fall into local optima solution due to its simplicity. Slow the convergence rate randomness is still a problem. Self-adaptability may be limitation under certain problems. Low efficiency, less accuracy can be experienced while dealing with multi-peak function.

3.8.5 Applications

Cuckoo search optimization algorithm applied for different problems in various domains. Power generations to minimise the cost of flues , n power with probability to generate in different values, Cloud computing security frameworks are-Gathering information, Network mapping, vulnerabilities exploration, audits and penetration tests, vulnerabilities enumeration and categorization, technology selection for vulnerability remediation, security solutions implementation. The security technology is used to decrease the vulnerability and costs are called Set covering problem [75] that is the Distribution systems will have more power loss and poor voltage regulation and voltage stability. VANET protocols design [76], electromagnetic and antenna arrays [77], classification of IDS [78]. Self-adaptive algorithm for search accuracy of the CSA [79], Compression factor to build [80], dynamic appropriate step-size [81]. CSA have been applied in many researchers in different application problems such as multilevel image thresholding, flood forecasting, wireless sensor networks, data fusion, cluster in wireless networks, clustering, ground water expedition, supplier selection, load forecasting, surface roughness identification, DG allocation in network, BPNN neural network, web service composition, speaker recognition, face recognition, training neural networks [82–85].

3.9 Moth flame optimization (MFO)

Mirjalili proposed moth flame optimization algorithm a swarm algorithm inspired by movement of moths in spiral path around light source. Moth flames randomly start searching in solution space. The fitness value estimated based on position by each moth in group. Falling category to best position flame by all is optimal solution. The function category updates following spiral movement function to achieve better division towards light source. The best position can be individual positions and repeats updating moth's distance and position generate new position to terminate criteria to be met. The variations in moth flame design in oder to improve are for multi-objective, binary and hybridization

3.9.1 Concept

Mirjalili proposed meta-heuristic algorithm based on population. MFO moths randomly with in space recognize fitness value and identify position suitable without flame. The movement is continuous and repeated to recognize better position. Update position suitably until termination criteria is met. The process MFO is carried on in three main steps. In first step, initialization of population and parameters are assumed in hyper dimensional space. The difference in way updates and treats in iterations. The position of each moth is stored. The selection of best moth is also performed so that results are stored longer time. In second step, three main functions converge to global result in Eq. (46). The identification to optimization is implemented randomly. Movement is spiral in moths applying logarithmic spiral function by Eq. (47). Moth and flame fixed position and indicate $[-1, 1]$ ranges. It balances between exploitation and exploration to guarantee moths circulation in search space guarantee in spiral motion. The fly of moth is traps of the local optima. Moth positioned near flame represented in matrix. In step 3, number of flames is updated; Moths locations search the exploitation in search space. Decrease and solve issue based on Eq. (48).

$$M(i,j) = (ub(i) - lb(j) * rand() + lb(i)) \tag{46}$$

$$S\left(M_i, F_j\right) = D_i e^{bt}.\cos(2\pi t) + F_j \tag{47}$$

$$flamecount = |N - l * \frac{N-l}{T}| \tag{48}$$

3.9.2 Algorithm and flowchart

The algorithm and flow of operations of MFOA is presented in **Figure 11**.

3.9.3 Merits

MFO similar to most population-based algorithm flexible and robust. The local opitma for individuals is avoided. Construction is easy and flexible in design. Moth has been incorporated to solve many engineering problems.

3.9.4 Demerits

Convergence is major issue in MFO.

3.9.5 Applications

MFO advantages have been incorporated in many domains. Navigation approach to solve the inequality and equality constrained optimization are real problem, to optimize real function for constrained selected variables. Chemical identification to improve single level production which can be extended to incorporate as include in determination of optimal production portfolio in other industries, applied in agriculture based to recognize problems of tomato [52]. Applied for medical field to improve time consuming Alhemeris disease, detection and diagnosis of breast cancer, to train networks RBFN [42], deployment of Wifi, determination of optimal solution in placement, location problem solution.

```
Algorithm : Moth-flame optimization Algorithm
Input: Number of moth-flame
Output: Best optimized solution
Begin
Parameter math-flam initialization
Moth position Pi randomly
For n moths do
Calculate fitness of all moths
end for
While not termination condition do
update position of Pi
calculate flames number
evaluate fitness function
if  initial round then
sort all moths & store best and second best
else
sort excluding previous best moth end if
For n moths do
For d flames do
Update parameters of moth flame
compute number of flame near to each moth and update respectively
end for
end for
end while
store best solution
end
```

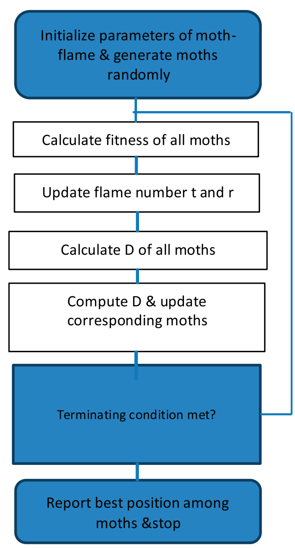


Figure 11.
MFO algorithms & flowchart.

3.10 Grey wolf optimization (GWO) algorithm

Grey wolf optimization, a meta-heuristic swarm technique, introduced for first time by Simon Fong [86]. Hunting behaviour in pack of wolf inspired in design of grey wolf optimization. Wolves in pack will not communicate physically during hunting, each wolf identify and attack prey individually silently. They follow levy flights model in search of food during hunting. Wolves unify to another pack of wolves or to new location if they find new food location better and suitable compared to their current dwelling place. A random hunter will be selected among pack to hunt for prey. The hunter identifies potential position itself to catch prey from current line of sight.

3.10.1 Concept

The social hierarchy consists of four levels in GWO. The level one called Alpha. They are the leaders of the pack, and they are male and female. They are responsible for making decisions about hunting time to walk, sleeping place and soon. The pack members have to dictate the alpha decisions and they acknowledge the alpha by holding their tails down. The alpha wolf is considered the dominant wolf in the pack and all his/her orders should be followed by the pack members. Next level group is labeled as Beta. The betas are subordinate wolves, which help the alpha in decision making. They can be either male or females. If consider the best candidate to both alpha when the alpha passes away or becomes very old. The beta reinforces the alpha's commands throughput the pack and gives the feedback to alpha. The third group of wolves is called Delta. They are subordinates. They need to submit their work report to alpha and beta. Scouts are responsible for watching boundaries of the territory and warning the pack in case of any danger. Sentinels are responsible for protecting the pack. Hunters are response got helping the alphas and beta involves beta in hunting and provide food for the pack. They are not important individuals in the pack, and they are allowed wolves were outwards. They are fighting i the case of loss.

Wolf search has been used to select two relay nodes: inter and intra relay nodes. Within a cluster, cluster members sense and transmit sensed data directly to the CH irrespective of their distance from CH. Hence, the nodes far away from CH dissipate more energy resulting in reduced network lifespan. To overcome this problem, the Wolf search is used in order to identify intra relay nodes for every cluster. The cluster member will send the sensed data to intra relay node and it in turm to CH. Similarly, all CHs communicate directly to BS irrespective of distance between CHs and BS. Hence, the CHs far away from BS dissipate more energy which leads to selection of new CHs resulting in next iteration, resulting very low network lifespan. To overcome this PEGASIS protocol introduced inter relay node as final node to communicate with BS. In proposed work, Wolf search is used to identify the inter relay nodes. The working principles of Wolf search for identification of inter and intra relay nodes are described in this section. The pseudo code of Wolf search is described in Algorithm 2.11.2.

(X, Y) are the coordinates of unknown node/target node and (x_i, y_i) are the coordinates of the i^{th} anchor node in the neighbourhood. The computations of WS) for encircling, and hunting process are shown below.

Eqs. (49)–(54) used in WSO are as follows.

$$d_i = \sqrt{(X - x_i)^2} + \sqrt{(Y - y_i)^2} \tag{49}$$

$$D = |C * X_p(t) - X(t)| \tag{50}$$

$$C = 2 * r \tag{51}$$

$$A = 2 * a * r - a \tag{52}$$

$$X(t+1) = X_p(t) - A * D \tag{53}$$

$$r = 0.5 + \frac{sin\, 2\sqrt{x^2 + y^2 - 0.5}}{(1 + 0.001X(x^2 + y^2))^2} \tag{54}$$

where t represents the current iteration, A and C are coefficient vectors, position vector of the prey is represented X_p, X the position vector, | | is the absolute value, and is an element-by-element multiplication, a is linearly decreased from 2 to 0 in each iteration and r is a random vector in [0, 1].

3.10.2 Flowchart and algorithm

The algorithm and flow of operation of GWO is presented in **Figure 12**.

Algorithm : Grey wolf optimization Algorithm
Input:
Output:
Begin
Initialize population of n candidate solutions Xi(I = 1, 2, 3……n)
Initialize α->best agent, β-> second best agent, δ->third best agent
While termination condition not met do
For each candidate wolf do
Update value of current candidate solution
End for
Update α β δ
Calculate fitness value
End while
end

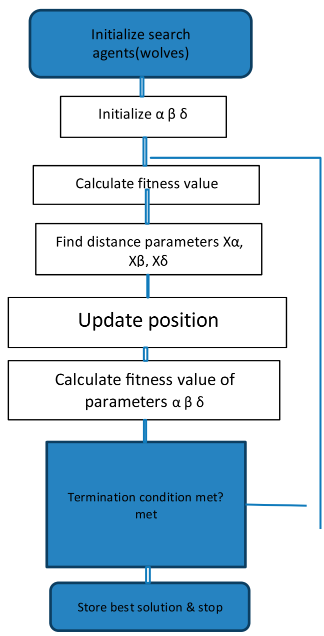


Figure 12.
GWO algorithm & flowchart.

3.10.3 Merits

GWO experience alpha and beta experience is good for complex problems. More heads identification and building are experienced wolves is built. The goodness is identified better through GWO. It is not easy to apply local optima compared to meta-heuristic has lesser parameters. It searches in local search space. The convergence is faster. GWO is easy to implement in any platform. In more iteration avoids local optima and provide higher performance in search problems.

3.10.4 Demerits

In short problem slow convergence and easy premature can be expected. Piece-wise linear cost approximation and update of equation to build local exploration ability are problem in GWO. The accuracy solving can be considered a research challenge. Bad local search ability and slow velocity and falling optimum behaviour and position update are required. Easy falling of velocity and fall into local optimum.

3.10.5 Applications of wolf-based algorithm

GWO algorithm finds application adaption in different domains. Fault system estimation and prediction, hydro-power optimal operation station, Optimization in multi-layer perception. Electronics based domain to find optimal allocation to determine system power loss, link functional net construction by q-Gaussian radial basis, Control operation of DC motors. Fault detection in power systems. Prioritization of problem, selection problem, solve combined economic emission dispatch problem to find optimum allocation. Multi-input and multi-output contingency management problem. Multi-input multi-output contingency management problems and for detection of faulty sections in power systems, to name a few.

4. Comparison of algorithms

Literature Survey reveals complex problems can be resolved in simple steps by applying bio-inspired principles and rules effectively by giving importance to each relationship. The discussed social and population based ten algorithms are involved in processing stages they include,

i. Identification of natural behaviour and responses of biological organism

ii. Replica model to simulate behaviour of biological organism

iii. Translating developed model to mathematical model with certain required assumptions

iv. Pseudocode generation for behaviours of biological organism

v. Experimenting practically and theoretically both models of biological organism for guaranteed performance improvements in real-world problem.

5. Issues, challenges and future direction

This section briefs on bio-inspired algorithm current challenges, issues and further direction for next works to be carried in this direction.

5.1 Literary issues

The database identification was first challenge to identify supporting literature. Scopus a largest database of academic articles was primary focus in collection of articles from journals. The published articles on specific bio-inspired algorithm searched for publication number from 2008 to 2020. During search process name was considered as keyword. Obtained results were analysed for algorithm, document-wise and subject wise. Documents are categories are article, conference paper, review, book and others. More research publication in different categories can be found based on bio-inspired algorithms PSO and GBA compared to other emerging algorithms which is plotted in **Figure 13**. To have clear view of published article year wise plot was plotted as shown in **Figure 14**. The evolution of algorithms is clearly shown in **Figure 14**. More publications have established algorithm PSO and GBA and on other remaining algorithms publications are comparatively low hence more research can be carried on to identify suitable optimization position for this algorithms.

5.2 Challenges

Bio-inspired algorithms face challenges in design of competitive and interactive component design. Biological systems have found lack in information exchange so algorithm has to be developed in absence of data. Improve or develop bio-inspired algorithms to design solution to adapt for any real-world problem. Performance of bio-inspired algorithm is another issue which need to be sorted in working environment.

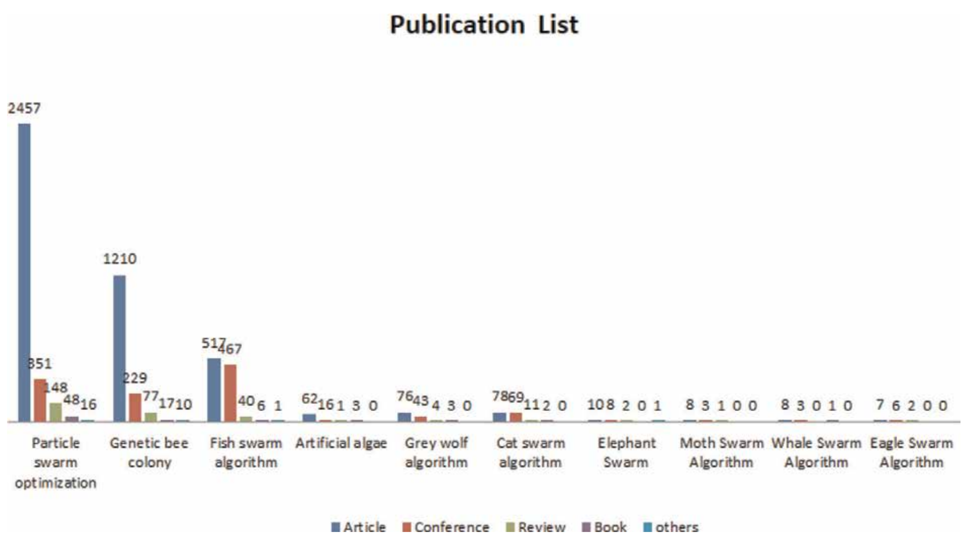


Figure 13.
List of publications on different bio-optimization.

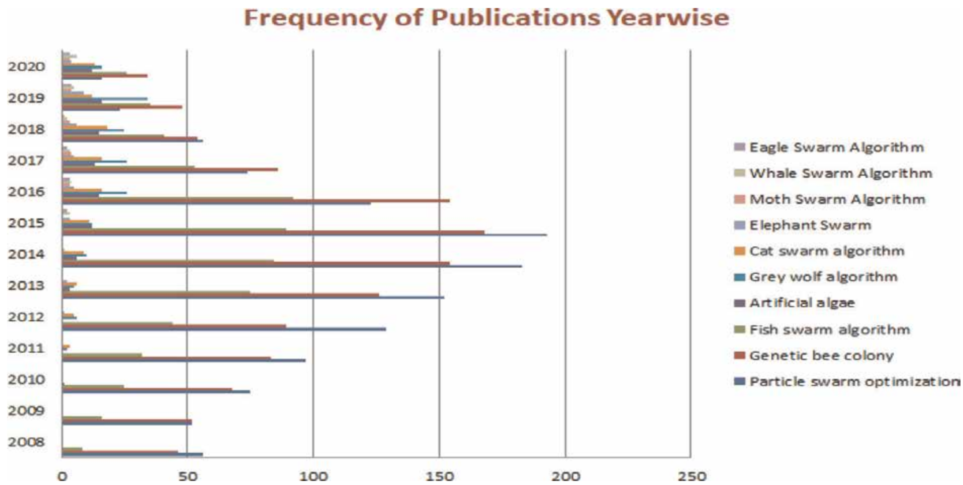


Figure 14.
Frequency of research articles published on various based algorithms till 2021 bio-inspired algorithms year wise.

5.3 Future scope

Bio-inspired algorithms brought revolutionary changes in different domains as well got power to impact further generation computing. The application coverage area is vast compared conventional methods includes modeling, algorithm, engineering and computing. Generally, optimization techniques based on swarm search procedures incorporate random changes and identification and still has capacity grow which is attracting many young researchers. Bio-inspired algorithms still require addressing new technologies along with it by exploring new ways to adopt algorithms. In order to achieve they need to be collaborated with research communities like computer science, biology, artificial intelligence, ecology, quantum and others. Currently, many bio-inspired algorithms exist, and application field is also extensive and obviously work require further exploration,

- -solution for specific application suitability of selection of parameters
- optimization in range and value of parameters.
- theoretical analysis of convergence of algorithm
- new application of bio-inspired algorithms needs to be explored
- identify suitable hybridization of algorithms with function or algorithms either convention or bio-inspired based.

6. Conclusion

Bio-inspired algorithms have got roots in both pure science and engineering domains. Methods and related theories are mature got huge practical potential benefits to provide in different domain problems. To conclude ten bio-inspired algorithms, FSA imitates food search behaviour of fish considering three parameters distance, length, crowd factor among them first two influence function much. WOA whale inspired algorithm has three operators applied to model search, encircle and foraging behaviour of whales. CSO includes two key operations seeking and tracing in computation of optimum solution. AAA control parameters influence whole functionality. MFO accuracy is based on spiral movement towards artificial light. ESA is based on exploration and exploitation in searching. GWO algorithm simulates wolves by dividing into four group; alpha, beta, delta, and omega.

Author details

G. Devika[1*] and Asha Gowda Karegowda[2]

1 Government Engineering College, K. R. Pet, Karnataka, India

2 Siddaganga Institute of Technology, Tumkur, Karnataka, India

*Address all correspondence to: sgdevika@gmail.com

References

[1] Mishra KK, Tiwari S, Misra AK. Bio inspired algorithm for solving optimization problems. In: 2011 2nd International Conference on Computer and Communication Technology (ICCCT-2011). Allahabad, India: Motilal Nehru National Institute of Technology; 2011. pp. 653-659. DOI: 10.1109/ ICCCT.2011.6075211

[2] Hammod WA, Zamil K, Ali A. A review of bio-inspired algorithms. In: A Conference: (SOFTEC Asia 2017). Malaysia: Kuala Lumpur Convention Centre; 2017

[3] Das S, Suganthan PN. Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation. 2011;**15**(1):4-31

[4] Fan X, Sayers W, Zhang S, et al. Review and classification of bio-inspired algorithms and their applications. Journal of Bionic Engineering. 2020;**17**: 611-631

[5] Kar AK. Bio-inspired computing—A review of algorithms and scope of applications. Expert Systems with Applications. 2016;**5**:20-32

[6] Swarm intelligence based energy efficient clustering algorithms for WSN: A review of algorithms, analysis and scope of applications. Chapter 5, Resource optimization using swarm intelligence and the IoT

[7] Devika G, Ramesh D, Karegowda AG. Analysis of binary and grey wolf optimization algorithms applied for enhancing performance of energy efficient. In: IEEE Proceedings. M S Raimaiah Institute of Technology; 2019

[8] Eftimov T, Korošec P, Koroušić Seljak B. A novel approach to statistical comparison of meta-heuristic stochastic optimization algorithms using deep statistics. Information Sciences. 2017; **417**:186-215

[9] Eftimov T, Korošec P, Koroušić Seljak B. Comparing multi-objective optimization algorithms using an ensemble of quality indicators with deep statistical comparison approach. In: Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI 2017). Hawaii, USA; 2017

[10] Gourtani A, Nguyen TD, Xu H. A distributionally robust optimization approach for two-stage facility location problems. EURO Journal of Computation Optimization. 2020;**8**:141-172

[11] Jabeen H, Jalil Z, Baig AR. Opposition based initialization in particle swarm optimization. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers; New York, NY, USA. 2009. pp. 2047-2052

[12] Zhang C et al. A novelswarm based Quasai oppositional PSO. In: 5th International Forum of Information Technology and Applications. 2009. pp. 325-330

[13] Clarccle C, Kennedy J. A particle swar-explosive, stability and convergence in multi-dimensional complex space. IEEE Transactions on Evolutionary Computation. 2002;**6**:58-73

[14] Imran M, Jabeen H, Ahmad M, Ababs Q, Bangyal W, Ababs Q. Opposition based PSO and mutation operators. In: 2nd International Conference on Education Technology

and Computer. Shanghai; 2010. pp. 506-508

[15] Zhang L, Yu H, Hu S. A new approach to improve particle swarm optimization. In: Proceedings of the 2003 International Conference on Genetic and Evolutionary Computation. 2003. pp. 134-139

[16] Weiand Wang J. A dynamical particle swarm optimization with dimensional mutation. IJCSNS International journal of computer Science and Network Security. 2006;**6**: 221-224

[17] Wang H. A hybrid PSO with Cauchely mutation. In: Proceeding of IEEE Transactions on Swarm Intelligence Symposium. Honolulu, HI, USA; 2007. pp. 356-360

[18] Imran M, Manzoor Z, Ali S, Ababs Q. Modified particle swarm optimization with student T mutation. In: International Conference on Computer Networks and Information Technology. Abbottabad; 2011. pp. 283-286

[19] Hajihassani M, Jahed Armaghani D, Kalatehjari R. Applications of particle swarm optimization in geotechnical engineering: A comprehensive review. Geotechnical and Geological Engineering. 2018;**36**:705-722

[20] Ma R-J, Yu N-Y, Hu J-Y. Application of particle swarm optimization algorithm in the heating system planning problem. Scientific World Journal. 2013; **2013**:718345

[21] Karaboga D. An Idea Based on Honey Bee Swarm for Numerical Optimization. Erciyes: Computer Engineering Department, Erciyes University; 2005

[22] Alshamlan HM, Badr GH, Alohali YA. Genetic Bee Colony (GBC) algorithm: A new gene selection method for microarray cancer classification. Computational Biology and Chemistry. 2015;**56**:49-60

[23] Feng X, Yin J, Xu M, Zhao X, Wu B. The algorithm optimization on artificial fish-swarm for the target area on simulation robots. In: IEEE 2nd International Conference on Signal Processing Systems (ICSPS). Zhenjiang, Jiangsu, China; 2010. pp. 87-89

[24] Alam F, Saadi HS, Alam MS. A novel comparative study between dual population genetic algorithm and artificial bee colony algorithm for function optimization. In: 2016 19th International Conference on Computer and Information Technology (ICCIT). Dhaka; 2016. pp. 333-338

[25] Xiao L. A clustering algorithm based on artificial fish school. In: 2nd International Conference on Computer Engineering and Technology (ICCET). Chennai, India: Contemporary Engineering and Technology; 2010. pp. 766-769

[26] Zhang M, Cheng S, Li F, Gan Y, Sun J. Evolving neural network classifiers and feature subset using artificial fish swarm. In: IEEE International Conference on Mechatronics and Automation. China; 2006. pp. 1598-1602

[27] Zhuang D, Ma K, Tang C, Liang Z, Wang K, Wang Z. Mechanical parameter inversion in tunnel engineering using support vector regression optimized by multi-strategy swarm algorithm. Tunnelling and Underground Space Technology. 2019;**83**:425-436

[28] Yazdani D, Nasiri B, Sepas-Moghaddam A, Meybodi M, Akbarzadeh-Totonchi M. mNAFSA: A

novel approach for optimization in dynamic environments with global changes. Swarm and Evolutionary Computation. 2014;**18**:38-53

[29] Kang C, Wang S, Ren W, Lu Y, Wang B. Optimization design and application of active disturbance rejection controller based on intelligent algorithm. IEEE Access. 2019;**7**:59862-59870

[30] Feng Y, Zhao S, Liu H. Analysis of network coverage optimization based on feedback k-means clustering and swarm algorithm. IEEE Access. 2020;**8**:42864-42876

[31] Yan L, He Y, Huangfu Z. A fish swarm inspired holes recovery algorithm for wireless sensor networks. International Journal of Wireless Information Networks. 2020;**27**(1):89-101

[32] Du T, Hu Y, Ke X. Improved quantum artificial fish algorithm application to distributed network considering distributed generation. Computational Intelligence and Neuroscience. 2015;**2015**:91

[33] Li X, Keegan B, Mtenzi F. Energy efficient hybrid routing protocol based on the artificial fish swarm algorithm and ant colony optimisation for wsns. Sensors. 2018;**18**(10):3351

[34] Kalaipriyan T, Amudhavel J, Sujatha P. Whale optimization algorithm for combined heat and power economic dispatch. Advances and Applications in Mathematical Sciences. 2017;**17**(1):197-211

[35] He N, Belacel A, Chan H, Hamam Y. A hybrid artificial fish swarm simulated annealing optimization algorithm for automatic identification of clusters. International Journal of Information Technology & Decision Making. 2016;**15**(05):949-974

[36] Sathya DJ, Geetha K. Hybrid ANN optimized artificial fish swarm algorithm based classifier for classification of suspicious lesions in breast DCE-MRI. Polish Journal of Medical Physics and Engineering. 2017;**23**(4):81-88

[37] Fei T, Zhang L. Application of BFO-AFSA to location of distribution centre. Cluster Computing. 2017;**20**(4):3459-3474

[38] Mao M, Duan Q, Duan P, Hu B. Comprehensive improvement of artificial fish swarm algorithm for global mppt in pv system under partial shading conditions. Transactions of the Institute of Measurement and Control. 2018;**40**(7):2178-2199

[39] Yan W, Li M, Pan X, Wu G, Liu L. Application of support vector regression cooperated with modified artificial fish swarm algorithm for wind tunnel performance prediction of automotive radiators. Applied Thermal Engineering. 2020;**164**:114543

[40] Zhu Y, Xu W, Luo G, Wang H, Yang J, Lu W. Random forest enhancement using improved artificial fish swarm for the medial knee contact force prediction. Artificial Intelligence in Medicine. 2020;**103**:101811

[41] Serapiao AB, Correa GS, Goncalves FB, Carvalho VO. Combining K-means and K-harmonic with fish school search algorithm for data clustering task on graphics processing units. Applied Soft Computing. 2016;**41**:290-304

[42] Pei-Wei T, Pan J-S, Chen S-M, Liao B-Y, Hao S-P. Parallel cat swarm optimization. In: International Conference on Machine Learning and Cybernetics. 2008. pp. 3328-3333

[43] Liu Y, Wu X, Shen Y. Cat swarm optimizing clustering: A cat swarm optimization in advances in swarm intelligence. In: Lecture Notes in Computer Science (LNCS 6782). 2011. pp. 321-328

[44] Saha SK, Ghoshal SP, Kar R, Mandal D. Cat Swarm Optimization Algorithm for Optimal Linear Phase FIR Filter Design. ISA Transactions in Press, Elsevier Ltd; 2013

[45] Abdel-Basset M, El-Shahat D, El-henawy I, Sangaiah AK, Ahmed SH. A novel whale optimization algorithm for cryptanalysis in merkle-hellman cryptosystem. Mobile Networks and Applications. 2018;**23**(4):723-733

[46] Ong P. Adaptive cuckoo search algorithm for unconstrained optimization. Scientific World Journal. 2014;**2014**:1-8

[47] Li-Jiang Y, Tian-Lun C. Application of chaos in genetic algorithms. Communications in Theoretical Physics. 2002;**38**:168

[48] Xu Z, Yu Y, Yachi H, Ji J, Todo Y, Gao S. A novel memetic whale optimization algorithm for optimization. In: International Conference on Swarm Intelligence. Cham, Switzerland: Springer; 2018

[49] Pecora LM, Carroll TL. Synchronization of chaotic systems. Chaos: An Interdisciplinary Journal of Nonlinear Science. 2015;**25**:097611

[50] Naik MK, Panda R. A novel adaptive cuckoo search algorithm for intrinsic discriminant analysis based face recognition. Applied Soft Computing. 2016;**38**:661-675

[51] Thi-Kien D, Pan T-S, Pan J-S. A multi-objective optimal mobile robot path planning based on whale optimization algorithm. In: Signal Processing (ICSP), 2016 IEEE 13th International Conference on IEEE. 2016

[52] Faseela CK, Vennila H. Economic and Emission Dispatch using Whale optimization Algorithm (WOA). In: 2018 in IJECE. 2018

[53] Fong M. A multi-objective optimal vehicle fuel consumption based on whale optimization algorithm. In: Advances in Intelligent Information Hiding and Multimedia Signal Processing: Proceeding of the Twelfth International Conference on Intelligent Information Hiding and Multimedia Signal Processing. Kaohsiung, Taiwan; 2016, 2016

[54] Marler RT, Arora JS. Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization. 2004;**26** (6):369-395

[55] Yan Z, Sha J, Liu B, Tian W, Lu J. An ameliorative whale optimization algorithm for multi-objective optimal allocation of water resources in Handan, China. Water. 2018;**10**:87

[56] Mafarja MM, Mirjalili S. Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. Neurocomputing. 2017;**260**: 302-312

[57] Trivedi IN, Jangir P, Kumar A, Jangir N, Totlani R. A novel hybrid PSO–WOA algorithm for global numerical functions optimization. In: Advances in Computer and Computational Sciences. Cham, Switzerland: Springer; 2018

[58] Jadhav AN, Gomathi N. WGC: Hybridization of exponential grey wolf optimizer with whale optimization for

data clustering. Alexandria Engineering Journal. 2017;**57**(3):1569-1584

[59] Revathi ST, Ramaraj N, Chithra S. Brain storm-based Whale Optimization Algorithm for privacy-protected data publishing in cloud computing. Cluster Computing. 2018;**21**:1-10

[60] Prakash D, Lakshminarayana C. Optimal siting of capacitors in radial distribution network using Whale Optimization Algorithm. Alexandria Engineering Journal. 2017;**56**(4):499-509

[61] Touma HJ. Study of the economic dispatch problem on IEEE 30-bus system using whale optimization algorithm. International Journal of Engineering, Science and Technology. 2016;**5**(1):1

[62] Tirkolaee EB, Alinaghian M, Hosseinabadi AAR, Sasi MB, Sangaiah AK. An improved ant colony optimization for the multi-trip Capacitated Arc Routing Problem. In: Computers & Electrical Engineering. 2018

[63] Kaveh A, Ghazaan MI. Enhanced whale optimization algorithm for sizing optimization of skeletal structures. Mechanics Based Design of Structures and Machines. 2017;**45**(3):345-362

[64] Oliva D, El Aziz MA, Hassanien AE. Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm. Applied Energy. 2017;**200**:141-154

[65] Gupta S, Saurabh K. Modified artificial killer whale optimization algorithm for maximum power point tracking under partial shading condition. In: Proceedings of the 2017 International Conference on Recent Trends in Electrical, Electronics and Computing Technologies (ICRTEECT). Warangal, Telangana, India; 2017. pp. 87-92

[66] Mostafa A, Hassanien AE, Houseni M, Hefny H. Liver segmentation in MRI images based on whale optimization algorithm. Multimedia Tools and Applications. 2017;**76**(23):24931-24954

[67] Karaboga D, Akay B, Ozturk C. Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. MDAI. 2007;**7**:318-319

[68] Rakitianskaia AS, Engelbrecht AP. Training feedforward neural networks with dynamic particle swarm optimisation. Swarm Intelligence. 2012;**6** (3):233-270

[69] Wang L, Zeng Y, Chen T. Back propagation neural network with adaptive differential evolution algorithm for time series forecasting. Expert Systems with Applications. 2015;**42**(2): 855-863

[70] Aljarah I, Faris H, Mirjalili S. Optimizing connection weights in neural networks using the whale optimization algorithm. Soft Computing. 2018;**22**(1): 1-15

[71] Uymaz SA, Tezel G, Yel E. Artificial algae algorithm (AAA) for nonlinear global optimization. Applied Soft Computing. 2015;**2015**:153

[72] Liu CP, Ye CM. Solving permutation flow shop scheduling problem by firefly algorithm. Industrial Engineering Management. 2012;**17**:56-59

[73] Fernandez-Viagas V, Framinan JM. A bounded-search iterated greedy algorithm for the distributed permutation flowshop scheduling problem. International Journal of Production Research. 2015;**53**:1111-1123

[74] Mustafa İ, Salih D, Hamid T, Kenan K, Fatih M. Optimization of filter by using support vector regression machine

with Cuckoo search algorithm. Radioengineering. 2014;**23**:790-797

[75] Yang XS, Deb S. Cuckoo search: Recent advances and application. Neural Computing and Applications. 2014;**24**: 169-174

[76] G. Zheng, S. P. Mohanty, E. Kougianos, "Metamodel-assisted fast and accurate optimization of an OP-AMP for biomedical applications", in Proceedings of the 11th IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2012; 273–278

[77] Chifu VR, Pop CB, Salomie I, Suia DS, Niculici AN. Optimizing the semantic web service composition process using cuckoo search. Computational Intelligence. 2012;**382**: 93-102

[78] Yildiz AR. Cuckoo search algorithm for the selection of optimal machine parameters in milling operations. International Journal of Advanced Manufacturing Technology. 2012;**64**: 55-61

[79] Ahmed T, Obaidi S. Improved Scatter Search Using Cuckoo Search. International Journal of Advanced Research in Artificial Intelligence. 2013; **2**:61-67

[80] Naik MK, Maheshwari RN, Wunnava A, et al. A new adaptive cuckoo search algorithm. In: IEEE International Conference on Recent Trends in Information Systems. Kolkata; 2015. pp. 1-5

[81] Burnwal S, Deb S. Scheduling optimization of flexible manufacturing system using cuckoo search-based approach. International Journal of Advanced and Manufacturing Technology. 2012;**64**:954-959

[82] Wang M, Chen H, Yang B, Zhao X, Hu L, Cai Z, et al. Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses. Neurocomputing. 2017;**267**: 69-84

[83] Singh P, Prakash S. Optical network unit placement in fiber-wireless (fiwi) access network by moth-flame optimization algorithm. Optical Fiber Technology. 2017;**36**:403-411

[84] Shehab M, Khader AT, Laouchedi M. Modified cuckoo search algorithm for solving global optimization problems. In: International Conference of Reliable Information and Communication Technology. Springer; 2017c. pp. 561-570

[85] Hassanien AE, Gaber T, Mokhtar U, Hefny H. An improved moth flame optimization algorithm based on rough sets for tomato diseases detection. Computers and Electronics in Agriculture. 2017;**136**:86-96

[86] Faris H, Aljarah I, Mirjalili S. Evolving radial basis function networks using moth–flame optimizer. In: Handbook of Neural Computation. Vol. 28. Elsevier; 2017. pp. 537-550

Section 2

Trends and Applications

Chapter 3

Designing Artificial Neural Network Using Particle Swarm Optimization: A Survey

Pooria Mazaheri, Shahryar Rahnamayan and Azam Asilian Bidgoli

Abstract

Neural network modeling has become a special interest for many engineers and scientists to be utilized in different types of data as time series, regression, and classification and have been used to solve complicated practical problems in different areas, such as medicine, engineering, manufacturing, military, business. To utilize a prediction model that is based upon artificial neural network (ANN), some challenges should be addressed that optimal designing and training of ANN are major ones. ANN can be defined as an optimization task because it has many hyper parameters and weights that can be optimized. Metaheuristic algorithms such as swarm intelligence-based methods are a category of optimization methods that aim to find an optimal structure of ANN and to train the network by optimizing the weights. One of the commonly used swarm intelligence-based algorithms is particle swarm optimization (PSO) that can be used for optimizing ANN. In this study, we review the conducted research works on optimizing the ANNs using PSO. All studies are reviewed from two different perspectives: optimization of weights and optimization of structure and hyper parameters.

Keywords: particle swarm optimization, artificial neural network, swarm intelligence, optimization, evolutionary algorithms

1. Introduction

ANN has been considered as an intelligent universal mechanism of dealing with function approximation, optimal design, process estimation, and prediction, pattern recognition, and other applications. Because of ANNs adaptability over a range of problems that involve decision making in uncertain situations, it is very attractive and popular amongst researchers. An ANN with many layers between the input layer and output layer is called Deep Neural Network (DNN). A large DNN may have millions of parameters that result in its learning process can take several days or even a month and need powerful hardware facilities. Also, there are several challenges which are required to address. For instance, the selection of the parameters, the structure of the networks, the selection of the initial values and the selection of the learning samples. If

ANN is designed with suitable parameters, it can be a powerful tool and lead to reducing learning time, minimizing loss function and make our predictions as accurate as possible. At this time, optimizers come to our aid. The optimizer helps us to build a better model, to improve the training process and some of them prevent to get trap in local optima.

Various methods exist to optimize a NNs. Backpropagation (BP) is one of them and it is used for optimizing Neural Networks [1–5]. BP training algorithm has different forms such as Gradient Descent, Levenberg-Marquardt, Conjugate Gradient Descent, Bayesian Regularization, Resilient, and One-Step Secant [6, 7]. For these algorithms, computational and storage requirements are different, some of these are suitable for an approximation of function and others for recognition of pattern, but they have disadvantages in a way or another such as the size of NN and storage requirements associated with them.

Another method is meta-heuristic algorithms. The objective of meta-heuristic algorithms is to discover global or local optimal solutions that are optimal with low cost. Meta-heuristic algorithms generally rely on various agents such as particles, chromosomes, and fireflies, searching iteratively to discover the global optimum or local optimum. Meta-heuristic is a collective concept of a series of algorithms such as evolutionary algorithm like Genetic Algorithm (GA) [8], naturally inspired algorithms such as PSO [9], trajectory algorithm like Tabu search [10], and etc.

In this paper, the focus is on PSO which is a nature-inspired algorithm for global optimization which can be utilized for solving the black-box optimization problem. Particle swarm is based upon simulation of the behavior of a school of fish or flock of birds. The use of active communication in such schools or swarms is a key concept. PSO like a GA is an optimization tool based upon population (swarm).

The goal of the study is to survey the papers which use PSO for optimizing ANN based on optimizing weights and biases and optimizing hyper parameters. There are some other surveys in this field, optimizing NN with evolutionary algorithms [11, 12], conventional and metaheuristic approaches [13], but this study only focuses on optimization of NN using PSO. In this survey, we try to categorize the existing methods for optimizing NN with PSO and show the role of hybrid and non-hybrid methods in optimization NN with PSO. The paper is organized as follows: In Section 2, Background Review, the architecture of Artificial Neural Network is explained with the backward and forward path for the BP method. Next, a brief overview of the PSO and its implementation is explained. Section 3, presents a review of the previous research related to optimizing ANN using PSO based on two categorizations. Section 4 will review challenges and gaps and finally, Section 5 will draw the Conclusion.

2. Background review

In this section, ANN, PSO, and the learning process in ANNs are reviewed.

2.1 Artificial neural network (ANN)

ANNs is considered a type of computational intelligence that is inspired by biological human systems like the brain process information [14]. ANNs are learned by instance and are configured for specific types of applications and problems through a learning system [15]. One of the most widely applied NN models is BP Neural

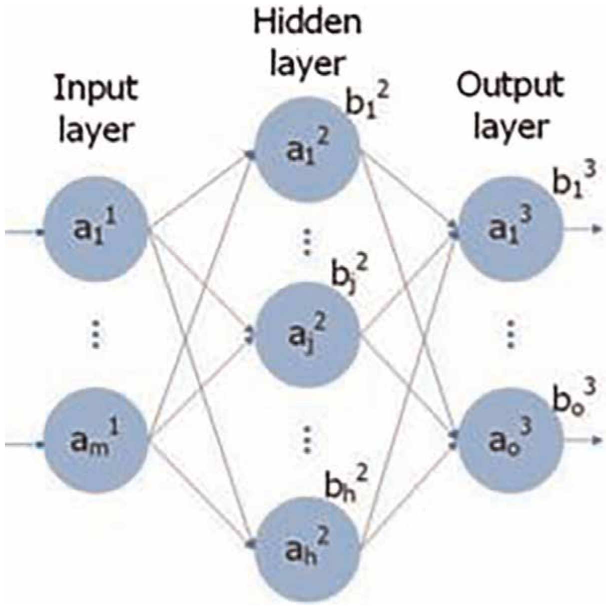


Figure 1.
Three-layer topological structure of BPNN.

Network (**Figure 1**). The framework of BP Neural Network is made of three kinds of layers, input layer, hidden layer, and output layer. The input layer and output layer are representatives of input variables and output variables, so that the number of input and output variables is equal to the number of neurons; depending on the specific problem, there may be one or more hidden layers. An ANN is called a Deep Neural Network when it is made up of more than three layers it means an input layer, multiple hidden layers, and an output layer. In different layers, the neuron junctions have their own weight, each output neuron is multiplied by a given weight and after summing up the result is used as the input to the next neuron. In the next step, the neurons generate the output signals by computations that are based upon the function of transfer, and then the gradient descent method is used to minimize the error function in order that the inferred network value be similar to the value of the target output as far as possible [16].

The learning process in a network consists of two steps: Feedforward (FF) and BP. The key principle is using the gradient descent method to minimize the error function and make a small change to the weights of the network [17].

The learning process is usually implemented in ANNs by instances; the learning process of ANNs has three types: supervised learning (SL), unsupervised learning (UL), and semi-supervised learning. The first type of learning process is SL that is based upon the direct comparison between the expected and actual output. The optimization algorithms are based upon gradient descent like BP algorithm, they can be used to iteratively modify the connection weights hence minimizing the error. UL is the second type that is based upon the correlation of the input data. The learning rule is the most important factor in the learning algorithm and can determine the weight update rules. Some popular learning rules are the Competitive Learning rule, Hebbian rule, and Delta rule [11]. The third type of learning process is semi-supervised learning. In this approach, a large amount of unlabeled data is combined with a small amount of labeled data. In fact, it can be said that semi-supervised learning falls between SL and UL.

2.2 Particle swarm optimization

The algorithm of PSO is used to optimize continuous nonlinear functions. It was proposed by J Kennedy and R Eberhart [18] and inspired by observations of collective and social behavior. PSO algorithm is considered a metaphor of social behavior. The social behavior is inspired by the movement of the flock to find food for the case of a bird flocking.

One of the advantages of PSO is the ability to deal with problems of multi-modal (i.e., multiple local optima) optimization and its simple implementation compared to associated strategies such as GA. PSO is used in various fields and has successfully been applied by several researchers to quantitative structure-activity relationship modeling, including kernel regression and k-nearest neighbor [19], minimum spanning tree for partial least squares modeling [20], piecewise modeling, and Neural Network training [21].

At first, the system will have a population of randomly created candidate solutions. Each candidate solution is called a particle, and it will throw into the problem space and will be given a random velocity. Each particle has memory and keeps track of previous corresponding fitness and best position. *pbest* call the previous best value. Therefore, *pbest* is associated only with a particular particle. The best value that exists between all the particles *pbest* in the swarm is *gbest*. The basic concept of the PSO technique is the acceleration of every particle toward its *pbest* and the *gbest* locations at every time step. Acceleration weights are random for both *gbest* and *pbest* locations. **Figure 2** indicates the concept of PSO. In this figure, P^k, P^{k+1}, V_{ini}, and V_{mod} are the current position, modified position, initial velocity, and modified velocity, respectively. V_{pbest} is velocity considering V_{gbest}, and *pbest* is velocity considering *gbest*.

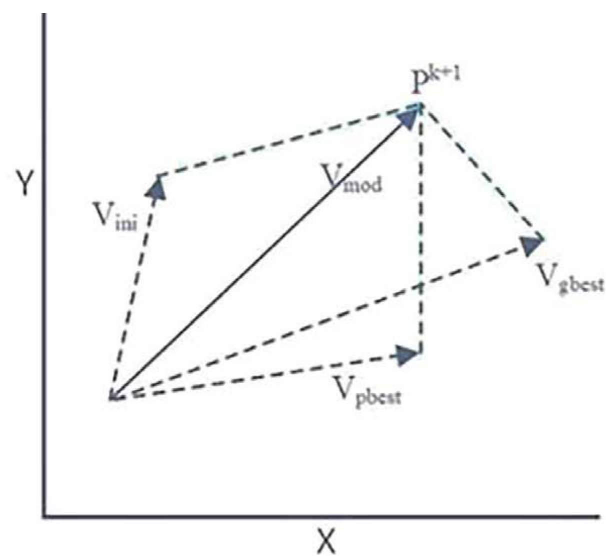


Figure 2.
Concept of changing a particle's position in PSO [22].

The PSO algorithm contains the following steps:

1. A population of particles initialized with random velocities and positions of d-dimension in the problem space.

2. Evaluate the desired optimization fitness function in terms of d variables for each particle.

3. *pbest* compare with particle's fitness evaluation. If *pbest* is worse than the current value then set *pbest* value and *pbest* location equal to the current value and the current location, respectively in d-dimensional space.

4. The population's overall previous best compare with fitness evaluation. When the *gbest* is worse than the current value then *gbest* changes to the current particle's array value and index.

5. Change the position and velocity of the particle according to Eqs. (1) and (2), respectively. $rand_1$ and $rand_2$ are two uniform random vectors. X_{id} and V_{id} also show the position and velocity of ith particle that has d-dimension, respectively.

$$V_{id} = (V_{id} * W) + (rand_1 * c_1 * (P_{bestid} - X_{id})) + rand_2 * c_2 * (G_{bestid} - X_{id})) \tag{1}$$

$$X_{id} = (V_{id} + X_{id}) \tag{2}$$

6. Step (2) is repeated until a criterion is met. This criterion usually is a maximum number of iterations or sufficiently suitable fitness calls.

PSO has several control parameters: W is the weight inertia that controls the exploitation and exploration of the search because it adjusts velocity dynamically. Asynchronous updates are less costly than synchronous updates. V_{max} is the largest velocity that is possible for the particles, if V_{max} is less than the velocity particle, the velocity of the particle decreases to V_{max}. Therefore, the fitness of search and resolution is directly affected by V_{max}. Particles are trapped in local minima when V_{max} is too low, and particles will move beyond good solution if V_{max} is too high. c_1 (cognition) and c_2 (social components) are the constants of acceleration. They change a particle velocity toward *gbest* and *pbest*. The tension is determined by velocity in the system. In a search space, a swarm of particles can be used globally or locally. In the PSO's local version, the entire procedure is the same and the *gbest* is replaced by the l_{best}.

3. The optimization of ANNs based on PSO algorithm

Methods with the aim of optimal design of an ANN utilizing PSO have been divided into two main categories: optimizing weights and optimizing structure and hyper parameters. These categories are further divided into two subcategories, including non-hybrid optimization and hybrid optimization, which in former authors only used PSO to optimize ANNs weights, in latter, hybrid methods have been utilized. Both subcategories have been reviewed in the following subsections A and B.

3.1 Weights and biases optimization

Some papers focused on weights for optimizing ANNs. They can be divided into two categories. First, those related to Non-Hybrid Optimization, and second, those related to Hybrid Optimization.

3.1.1 Non-hybrid optimization

Some studies used classical PSO to optimize NN and for showing their accuracy, they compared their solution with conventional approach optimization like BP. The first paper that falls into this category is from Gudise and Venayagamoorthy [23] published in 2003. They made a comparative study on the computational requirements of the BP and PSO algorithm for NN as training algorithms. They presented results for an FFNN learning a nonlinear function and indicated that the FFNN weights converge faster when the PSO is used instead of the BP algorithm. Later, in 2005, a modified PSO was presented by Zhao et al. [24], which adjusts the velocities and positions of the particle on the basis of the best positions that are earlier visited by other particles and themselves, and includes the method of diversifying the population to prevent premature convergence. In this paper, PSO is compared with the conventional BP to learn a nonlinear function for training a FFNN. The considered problem is how accurate and fast can the weights of NN be determined by BP and PSO to learn a common function. Another research that compared the PSO and BP for optimization NN was proposed by Ni et al. [25] in 2014. They introduced PSO for stochastic global optimization in NN training to solve the flaws of the traditional BP network in cementing prediction. They showed their method's training time is shorter than BP network and also the prediction accuracy that they obtained is high. Following by that, Liu et al. [26] to predict the high-speed grinding temperature used a BP NN based upon PSO algorithm (PSO-BP). They compared their method with gradient descent training BP NN which trained based upon Levenberg Marquardt (LM) algorithm and showed that PSO- BP performs better than the other methods in predicting the grinding temperature. In this paper, the authors used PSO algorithm for training BP NN to obtain a set of weights and biases, which could minimize the Mean Square Error (MSE).

In some studies, firstly PSO was improved and then used for optimizing NN. First, Bai et al. [27] used improved PSO- BP NN to improve the prediction accuracy of pest occurrence cycle. Their method used inertia weight to improve the PSO algorithm. Next, they used improved PSO to optimize the thresholds and weights of BP NN. Then, they established a pest prediction model using a rough set and an improved PSO- BP network. Their research showed that the number of iterations can be reduced by the improved PSO algorithm. Second, Liu and Yin [28] optimized BP NN with using an improved PSO. In the new algorithm, PSO used enhanced adaptive acceleration factor and also enhanced adaptive inertia weight to justify the initial weight value and biases of BP NN. At the end, simulation results indicated that the new algorithm is able to enhance convergence rate and precision of prediction of BP NN, that decreases the error of prediction. Later on, Nandi and Jana [29] rectified the problem by formulating a new inertia weight strategy for PSO called PPSO which balanced the exploitation and exploration properly while training ANN and compared their model with 4 other training algorithms. For all benchmark datasets, PPSO showed better performance with regard to avoiding local minima and convergence rate as well as better accuracy. The proposed PPSO reduced the trapping risk in local minima with a very well convergence rate.

In some works, PSO was employed to optimize NN in different fields such as medical imaging, energy consumption, civil engineering, etc. For example, in medical imaging, Wang et al. [30] introduced a method of relatively recent image enhancement for improving the brain image contrast. Then, they presented the Predator-Prey PSO (PP-PSO), which is a modification of traditional PSO to train weights of single-hidden layer NN. In their method, they utilized the MSE as an objective function. Later on, Zhang et al. [31] developed a technique that could automatically establish diagnoses from the brain magnetic resonance images. First, the processing brain imaging was implemented. Second, from the volumetric image, one axial slice was selected. Third, a single-hidden layer NN was utilized as a classifier. Finally, for training the weights and biases of the classifier, a predator-prey PSO was proposed. Their method performs better than the human observers and 10 state-of-the-art approaches. Also, in energy consumption, Le et al. [32] proposed four novel AI techniques. They utilized these models for predicting the heating load of buildings' energy efficiency. Their model was based upon meta-heuristics algorithms and the potential of ANN, including Imperialist Competitive Algorithm (ICA), Artificial Bee Colony optimization (ABC), GA, and PSO. For the buildings prediction of the heating load of energy efficiency with PSO-ANN model, the parameters of the PSO algorithm were set up before optimization of the ANN model consisting of the number of particle swarms, maximum particle's velocity, individual cognitive, group cognitive, inertia weight, and maximum number of iterations. Then, PSO algorithm optimized the biases and weights of the initialized ANN. The best PSO-ANN model was determined with the lowest Root Mean Squared Error (RMSE). The GA provided the highest performance in optimizing the ANN model, to forecast the HL of EEB systems. The remaining meta-heuristics algorithms provided more unsatisfactory performance, in contrast to the performance of the ICA-ANN, PSO-ANN, and ABC-ANN models. In the civil engineering field, Chatterjee et al. [33] proposed a PSO-based approach to train the NN for predicting structural failure of the reinforced concrete buildings. In order to find the optimal weights for the NN classifier, the PSO algorithm was involved. In the first phase, NN training, PSO minimizes the RMSE to achieve the optimal input weight vector to the input layer of the ANN. Next, to get ingenuity, the NN-PSO model was compared with MLP-FFN classifier (multilayer perceptron FF network) and NN. Finally, the supremacy of the presented NN-PSO in comparison to the NN and MLP-FFN classifiers was shown by the experimental results.

Besides, some studies have focused on only a specific version of NN like random FF NN (RFNN) and tried to use PSO to optimize them. For example, Xu and Shu [34] at the beginning, considered the advantages of both PSO and non-iterative learning to train RFNN. Pacifico and Ludermir [35] presented to utilize PSO and clustering analysis to optimize RFNN input weights and biases. In this study, they employed a local best neighborhood scheme for PSO population updating where each individual only followed some members belongs to its immediate neighborhood. Following by that, an improved PSO was proposed by Ling et al. [36], which encoded the input-to-output sensitivity information of RFNN to optimize the input weights and biases.

Some researchers to find a better answer for their problems, used different types of PSO such as cooperative PSO, Cultural Cooperative Particle Swarm Optimization (CCPSO), and multi-phase PSO. The cooperative PSO is an enhanced PSO that was presented by Van den bergh and Engelbrecht [37]. They obtained good results by applying this method on NN training. In this method, input vectors are divided into several sub vectors that are optimized in their own swarms cooperatively. In this case, performance is improved due to splitting the main vector into several sub vectors which in turn results in better credit assignments and decreases the chance to omit a possible good solution for a certain component in the vector. Lin et al. proposed [38] a CCPSO approach that a collection of multiple swarms which interact by exchanging information. They applied CCPSO for optimizing a fuzzy NN and result in it performed better than BP and GA. Next, Multi-phase PSO (MPPSO) was proposed by Al-kazemi and Mohan [39] in 2002. Training of ANNS by MPPSO is another variation which evolves simultaneously multiple groups of particles that change the direction of search in different phases of the algorithm. Each particle in this method is in a specific group and phase at a given time. MPPSO boosts the broader exploration of the search space, increases population diversity, and prevents premature convergences. Furthermore, MPPSO has different update equations comparing to the basic PSO and permits changes to the locations of the particle that only lead to some improvements. Many researchers chose a different path and have used multiobjective PSO for optimizing NN. For example, Carlos Coello et al. [40] proposed Multiobjective Particle Swarm Optimization (MOPSO) and used this method as a searching strategy for improving NN.

Some studies utilized PSO for solving large-scale problems. For instance, a novel study for high-dimensional datasets was proposed to optimize the weights of NN with PSO and some other Evolutionary Computation (EC) methods. Xue et al. [41] presented a self-adaptive parameter and strategy-based PSO (SPS-PSO) algorithm and then they used this method to optimize FFNN with feature selection. The authors divided the experiments into two groups. They utilized SPS-PSO and three other evolutionary computation methods, GA, PSO, and biogeography-based optimization for directly optimizing the FNN's weights in the first group. In the other group, firstly, they employed SPS-PSO-based feature selection on the initial datasets and obtained eight comparatively smaller datasets with the K-Nearest Neighbor (KNN). Then, the new datasets were utilized as the inputs for FNN. They optimized the FNN weights one more time by SPS-PSO and three other evolutionary computation methods. The experimental findings showed that SPS-PSO had the vantage to optimize the FNN weights in comparison to the other methods of EC. Meanwhile, the feature selection based upon SPS-PSO can decrease the size of solution and computational complexity, whereas ensuring the accuracy of classification, it is utilized for preprocessing the datasets for FNN.

3.1.2 Hybrid optimization

In this subcategory, authors used hybrid methods to optimize weights of ANNs.

Some studies combining GA and PSO for optimizing ANN's weights. For instance, in 2018, Anand and Suganthi [42] optimized ANN with using a hybrid algorithm of PSO and GA. Then, they used this model to enhance the measurement of electricity demand in India. Their model has higher performance and reliable accuracy than ANN-PSO or ANN- GA that are single optimization models. They used hyperbolic tangent and identity as activation function in hidden layer and output layer, respectively, the sum of squares as error function and mean absolute percentage as an indicator of the quality of prediction. PSO by using linear and quadratic regression models together, optimized the weights of socio-economic indicators and performs a search for the best fitted members that lessen the error. Also, Ma [43] developed a short time traffic flow prediction software on the basis of BP NN that could be used for predicting urban short-term traffic flow. The GA-based improved PSO was utilized for optimizing BP NN weight threshold to improve BP NN prediction accuracy. The results showed that this software could accurately and quickly predict the information of road traffic flow at the next moment, which could extremely reduce urban road traffic pressure. Next, Xiao et al. [44] proposed a new three-stage nonlinear ensemble model. In this model, three various types of NN based models, including elman network, generalized regression NN, and wavelet NN built by three non-overlapping training sets. The results of the study showed the ensemble ANNs-PSO-GA method enhanced the prediction performance over other linear combination and individual models.

In some works, researchers preferred combining PSO and wavelet to obtain a better answer. In 2015, Zhang et al. [45] with using Wavelet Entropy (WE) proposed a novel computer-aided diagnosis system to extract some features from Magnetic Resonance (MR) brain images, followed by FFNN with training method of a Hybridization of PSO and biogeography-based optimization (HBP), which combined the exploration ability of biogeography-based optimization and exploitation ability of PSO. They used MSE as an objective function to optimize weights with PSO. The proposed WE+HBP-FNN method obtain nearly perfect detection pathological brains in MRI scanning. Next, a novel hybrid approach called Switching PSO-Wavelet Neural Network (WNN) was proposed by Yang Lu et al. [46] in 2015 to enhance recognition accuracy in face recognition that is one of the important research problems in computer vision. They used the algorithm of the recently proposed Switching PSO (SPSO) for optimizing the weight parameters, translation factors, scale factors, and threshold in WNN. The proposed method, SPSO- WNN, has a higher learning ability and fast convergence speed than conventional WNN. Especially, for overcoming the difference between the local search and the global search, which facilitates jumping the local minimum, a velocity-updating equation depended on mode with Markovian switching parameters is presented in SPSO. They showed their method has a much better performance compared to PSO-WNN, GA-WNN, and WNN.

Following by that, some studies tried to use a hybrid model to propose better models compare to BP. Firstly, in 2008, Chen et al. [47] used a hybrid evolutionary algorithm that is based upon PSO and AFSA, also referred to as AFSA-PSO- parallel-hybrid evolutionary (APPHE) algorithm in FFNN training. They showed that FFNN training by the novel hybrid evolutionary algorithm compared to FFNN trained by Levenberg-Marquardt BP (LMBP) algorithm, show high stability toward the optimal position, satisfactory performance, convergent accuracy and converges quickly. In this research, both the output transfer function and the hidden transfer function were sigmoid function. Secondly, a hybrid crop classifier was presented by Zhang and Wu [48] for polarimetric synthetic aperture radar images in 2011. The feature sets included the cloude decomposition known as H/A/α decomposition, span image, and the gray-level co-occurrence matrix-based texture features. Then, Principle Component Analysis (PCA) reduced the features. Lastly, an FNN was built and trained by Adaptive Chaotic PSO (ACPSO). The results on flevoland sites showed the superiority of ACPSO to BP and adaptive BP.

Some works prefer to combine BP and PSO to make a hybrid model for optimizing weights of NN. In 2007, Zhang et al. [49] proposed a hybrid algorithm combining BP with PSO algorithm. For training the weights of FFNN, the hybrid algorithm can benefit from employing strong global searching and local searching ability of the PSOA and the BP algorithm, respectively. Firstly, in the PSOBP algorithm, a heuristic algorithm was adopted by them to give a transition from PSO to gradient descending search. Also, they gave three kinds of encoding strategy of particles and gave the different problem areas that every encoding strategy was actively used in. They showed that in terms of accuracy and convergent speed, the proposed hybrid PSOBP algorithm performs better than the adaptive PSO and BP algorithm. Following by that, in 2011, Yaghini et al. [50] proposed a hybrid improved opposition-based algorithm that is based upon PSO and GA (HIOPGA) methods and then compared BP algorithm with their method on several benchmark problems. In fact, their method combined ability of two algorithms. This algorithm began training using a particle population. During the algorithm iteration, when improved opposition-based PSO cannot improve some particles' position, a subpopulation of such NNs is created and sent to GA. Now, the HIOPGA can find better NN to replace in the population by utilizing the GA operators, mutation, and crossover. Also, Kartheeswaran and Durairaj [51] in 2017, for image reconstruction, presented the sequential and parallel data implementing the decomposition strategies on a PSO algorithm based ANN weights optimization. They utilized a hybrid algorithm combining BP with PSO algorithm. They used PSO with BP-ANN for optimizing the different parameters including hidden layer sizes, number of hidden nodes, and optimize the network connection's weights. In fact, this study, by optimizing the weights of connection, presented the application of a hybrid model for the reconstruction of Shepp-Logan head phantom image.

3.2 Optimizing structure and hyper parameters

In this category, there are a few papers that have focused on optimizing hyper parameters. There are two subcategories: first Non-Hybrid Optimization, second, Hybrid Optimization.

3.2.1 Non-hybrid optimization

In this subcategory, the authors used non-hybrid methods to optimize structure and hyper parameters.

In 2000, Zhang and Shao [52] were the first authors that presented a PSONN system for evolving network architecture and the weights of ANNs, alternately. They used evolved ANNs in modeling product quality estimator for a fractionator of the hydrocracking unit in the oil refining industry. Carvalho and Ludermir [53] proposed another study that was inspired by Zhang and Shaos methodology but introduces the weight decay heuristic in the weight adjustment process in an attempt to obtain more generalization control. They analyzed the use of the PSO for the optimization of architectures and weights of NN with the aim of the performance of better generalization by making a compromise between low training errors and low architectural complexity and utilized them for specific problems in the medical field that fall within benchmark classification category. The results that they obtained, showed that a PSO-PSO based method indicates an acceptable alternative for optimizing architectures and weights of NNs of MLP. Xue et al. [54] similar to Carvalho and Ludermir tried to optimize weight and architecture simultaneously. They found a variable- length PSO to optimize both the number of hidden nodes and input weights, simultaneously. Particles with various lengths which showed various network configurations can be solved with a new particle update strategy presented in this study.

Many researchers improved the algorithms themselves to optimize architecture. Here are some examples: Carvalho [55], proposed a PSO-PSO method, in which a PSO was employed for optimizing weights that were nested under another PSO which was employed to optimize the architecture of FNN by deleting or adding hidden nodes. Next, in 2009, Kiranyaz et al. [56] proposed a multidimensional PSO approach to construct FNN by utilizing an architectural space, automatically. Furthermore, the individuals in the swarm population have been designed in a way that it optimized both the weights and architecture of an individual in every iteration.

PSO for optimizing NN's architecture used by researchers in different areas and topics such as communication theory, civil and medical engineering. PSO has been utilized widely to address the optimization problems existing in communication theory. Das et al. [57] optimized ANN by using PSO for the problem of channel equalization in 2013. In this paper, they used PSO algorithm to optimize all the variables including network parameters and network weights. In fact, they used the PSO to optimize the number of input neurons, hidden neurons, the type of transfer functions, and the number of layers. The novelty in this paper is that they take care of suitable network topology. Extensive simulations proposed in this research showed that, as compared to other ANN-based equalizers as well as neuro-fuzzy equalizers, the proposed equalizer performs better in all noise conditions. An interesting application area of PSO is civil engineering. The application of an improved PSO technique was proposed by Asadnia et al. [58] for training an ANN to predict water levels for the Heshui Watershed. The results showed that the PSO-based ANNs performed better to predict the peak and low water levels compare with the LM-NN model. Additionally, IPSONN had a quicker convergence rate in comparison with CPSONN. In medical engineering, an adaptive CPSO was developed by Zhang et al. [59] to train the parameters of FFNN, with the purpose of accurate classification of magnetic resonance (MR) brain images. The classification accuracy of the presented technique was 98.75% on 160 images.

Many works used basic PSO to optimize NN's architecture. In a study by Chunkai et al. [60], in 2000, the network structure is adaptively adjusted and the PSO algorithm is applied to evolve the nodes of the NN with a specific generated structure. The techniques such as the combination of partial training and evolving added nodes are used to generate the desired architecture and then PSO is employed to evolve the nodes of the predefined structure. In another study in 2013, Wang et al. [61] used the BP NN to build an estimation model for the cost of plastic injection modeling parts to decrease the complication of the conventional procedures of estimating all the costs. They have made an estimation model for costs on the basis of the superior capability in forecasting and diagnosis for BP NN, and the capability of the great solution caused by PSO was utilized to get the parameters for BP NN, such as the number of hidden nodes and layers, initial weight, learning rate, hence learning and training for the network were made to perform better and be more precise. In this study, the sigmoid function was utilized as activation function and transfer function. In 2018, Qi et al. [62] presented a combination of ANN and PSO for forecasting the unconfined compressive strength of Cemented Paste Backfill (CPB). The authors used ANN for nonlinear relationships modeling and also utilize PSO for tuning the ANN architecture. In fact, in this work, PSO optimized the number of neurons and hidden layers. The findings indicated that PSO was efficient for optimizing the ANN architecture. Also, comparing the values of forecast UCS with experimental values indicated that the model of optimal ANN was very precise to predict the strength of CPB.

3.2.2 Hybrid optimization

In this subcategory, authors employed hybrid methods to optimize the structure and hyper parameters of NNs.

In a study, J Yu et al. [63] presented a new evolutionary ANN algorithm called IPSONet. This algorithm was based on an improved PSO. The improved algorithm utilized parameter automation strategy, mutations, crossover, and velocity reset- ting to enhance the performance of the classical PSO in fine-tuning of the solutions and global search. To solve the design problem of FFNN, the improved PSO was used by IPSONet. They used the improved PSO to evolve simultaneously weights and structure of ANNs by the evolutionary scheme and a specific individual representation. Next, researchers employed hybrid GA and PSO to optimize structure and hyper parameters to obtain a better answer. For example, Juang [64] in 2004, presented a modified PSO Hybrid of GA and PSO (HGAPSO) method that was employed to design NN. In this method, the individuals of the next generation are created not only by crossover and mutation operators but also by PSO. The upper half of the best performing individuals in a population are enhanced using PSO and the other half is generated by applying the crossover and mutations. Unlike GA, HGAPSO removes the restrictions of evolving the individuals within the same generation. In this article, the proposed method is another variation of PSO for fixed structure ANNs where only weights are adjusted.

4. Challenges and gaps

Particle Swarm Optimization is a heuristic optimization method that performs well for various optimization problems. But like other swarm intelligence-based optimization technique, PSO has some disadvantages including sensitivity to parameters, high computational complexity, slow convergence. The first reason is that PSO is unable to employ the crossover operator as utilized in genetic algorithm or differential Evolution. Therefore, the distribution of suitable information be- tween candidates is not at an essential level. Another factor can be the fact that PSO is unable to handle appropriately the relationship between exploration and exploitation, in fact, local search and global search, so it often converges to a local minimum quickly. One of the solutions that can address these problems is hybridization. Numerous optimization algorithms have been utilized for ANN optimization like GA that some of them can be seen in this paper. For future work, PSO can be hybridized with some of these optimization algorithms like GA, SA, TS, DE, ABC, and ACO to develop hybrid approaches in order to achieve better exploration ability.

Another challenge is that study of PSO for optimizing NN had great achievements but there is no in-depth research on theoretical aspects. So, we think it can be interesting to conduct another study of both the run-time and convergence properties of PSO for optimizing NN. In addition, there are not many works related to PSO implemented in parallel for optimizing NN. Thus, it can be a potential path for future research. Moreover, considering other Deep Learning.

Finally, stream data poses significant challenges in this area. In a non-stationary environment, like weather forecasting and stock-price market, data comes in the stream. So, it can be a good topic to design strategies for the dynamic training of NN using PSO.

5. Conclusion

ANN as a fertile approach to developing an intelligent information processing system has been introduced. Specifically, ANNs have been seen as a powerful tool in

No.	Author/Authors	Year	Optimization task	Types
1	Zhang and Shao [52]	2000	Optimizing weights and structure	Non-Hybrid
2	Chunkai et al. [60]	2000	Optimizing structure	Non-Hybrid
3	Al-kazemi et al. [39]	2002	Optimizing weights	Non-Hybrid
4	Gudise and Venayag [23]	2003	Optimizing weights	Non-Hybrid
5	Vandenbergh and Engelbrecht [37]	2004	Optimizing weights	Non-Hybrid
6	Coello et al. [40]	2004	Optimizing weights	Non-Hybrid
7	Juang et al. [64]	2004	Optimizing structure	Hybrid
8	Meissner et al. [65]	2005	Optimizing weights	Non-Hybrid
9	Zhao et al. [24]	2005	Optimizing weights	Non-Hybrid
10	Carvalho and Ludermir [66]	2006	Optimizing weights	Hybrid
11	Xu and Shu [34]	2006	Optimizing weights	Non-Hybrid
12	J Yu et al. [63]	2007	Optimizing weights and structure	Hybrid
13	Carvalho and Ludermir [55]	2007	Optimizing weights structure	Non-Hybrid
14	Carvalho [53]	2007	Optimizing weights and structure	Non-Hybrid
15	Zhang et al. [49]	2007	Optimizing weights	Hybrid
16	Lin et al. [38]	2008	Optimizing weights	Non-Hybrid
17	Chen et al. [47]	2008	Optimizing weights	Hybrid
18	Kiranyaz et al. [56]	2009	Optimizing structure	Non-Hybrid
19	Zhang et al. [59]	2010	Optimizing structure	Non-Hybrid
20	Zhang and Wu [48]	2011	Optimizing weights	Hybrid
21	Yaghini et al. [50]	2011	Optimizing weights	Hybrid
22	Zhao [67]	2012	Optimizing weights	Non-Hybrid
23	Wang et al. [61]	2013	Optimizing structure	Non-Hybrid
24	Armaghani et al. [68]	2013	Optimizing weights	Non-Hybrid
25	Das et al. [57]	2013	Optimizing weights and structure	Non-Hybrid
26	Pacifico and Ludermir [35]	2013	Optimizing weights	Non-Hybrid
27	Xue et al. [54]	2013	Optimizing weights and structure	Non-Hybrid
28	Asadnia et al. [58]	2014	Optimizing structure	Non-Hybrid
29	Xiao et al. [44]	2014	Optimizing weights	Hybrid
30	Bai et al. [27]	2014	Optimizing weights	Non-Hybrid
31	Ni et al. [25]	2014	Optimizing weights	Non-Hybrid
32	Yang Lu et al. [46]	2015	Optimizing weights and scale factors	Hybrid
33	Zhang et al. [45]	2015	Optimizing weights	Hybrid
34	Liu et al. [26]	2016	Optimizing weights	Non-Hybrid
35	Wang et al. [30]	2016	Optimizing weights	Non-Hybrid
36	Chatterjee et al. [33]	2016	Optimizing weights	Non-Hybrid
37	Liu and Yin [28]	2016	Optimizing weights	Non-Hybrid
38	Zhang et al. [31]	2017	Optimizing weights	Non-Hybrid

No.	Author/Authors	Year	Optimization task	Types
39	Kartheeswaran and Durairaj [51]	2017	Optimizing weights	Hybrid
40	Pradeepkumar and Ravi [69]	2017	Optimizing weights	Hybrid
41	Anand and Suganthi [42]	2017	Optimizing weights	Hybrid
42	Ling et al. [36]	2017	Optimizing weights	Non-Hybrid
43	Qi et al. [62]	2018	Optimizing structure	Non-Hybrid
44	Yang and Jiang [70]	2018	Optimizing weights	Non-Hybrid
45	Kong et al. [71]	2019	Optimizing weights	Non-Hybrid
46	Ma [43]	2019	Optimizing weights	Hybrid
47	Xue et al. [41]	2019	Optimizing weights	Non-Hybrid
48	Le et al. [32]	2019	Optimizing weights	Non-Hybrid
49	Chen et al. [72]	2019	Optimizing weights	Non-Hybrid
50	Nandi and Jana [29]	2019	Optimizing weights	Non-Hybrid

Table 1.
Optimization types that researchers used.

modern AI techniques. To utilize a prediction model based upon ANN, we face some challenges that ANN training is one of the major of them. For training ANN, conventional algorithms are used which results in researchers faced some problems. These conventional algorithms like backpropagation, are local search methods that exploit the current solution to produce a new solution. However, they lack exploration ability, hence, they often, finds local minima of an optimization problem. Unlike conventional approaches, metaheuristics like PSO are good at both exploration and exploitation and are able to solve simultaneous adaptation in each component of NN. In this paper, we present a survey of optimizing and training ANNs with using PSO that is one of the best metaheuristic algorithms for optimizing ANN. We try to review some studies conducted on optimizing ANN using PSO for different goals including comparing different methods results and solving various types of problems. In this study, all the papers are grouped into categories including the kind of PSO, year of publication, activation fitness function types, and what has been optimized. Findings in this study provide future direction for further work on optimizing ANN with using PSO (**Table 1**).

Author details

Pooria Mazaheri*, Shahryar Rahnamayan and Azam Asilian Bidgoli
Nature Inspired Computational Intelligence (NICI) Lab, Ontario Tech University, Ontario, Canada

*Address all correspondence to: pooria.mazaheri@ontariotechu.ca

References

[1] Werbos PJ. Backpropagation through time: What it does and how to do it. Proceedings of the IEEE. 1990;**78**(10): 1550-1560

[2] Werbos PJ. The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting. Vol. 1. John Wiley & Sons; 1994

[3] Rumelhart DE, Hinton GE, Williams RJ. Learning internal representations by error propagation Tech. Rep. 1985

[4] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature. 1986; **323**(6088):533-536

[5] Hagan M, Demuth H, Design MBNN. Boston, MA. 1996

[6] Hagan MT, Menhaj MB. Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks. 1994;**5**(6):989-993

[7] Chen C, Lai H. An empirical study of the gradient descent and the conjugate gradient backpropagation neural networks

[8] Goldberg DE, Holland JH. 1988

[9] Tian Z, Fong S. Optimization algorithms methods and Applications. 2016

[10] Glover F. Tabu search—part I. ORSA Journal on Computing. 1989;**1**(3): 190-206

[11] Ding S, Li H, Su C, Yu J, Jin F. Evolutionary artificial neural networks: A review. Artificial Intelligence Review. 2013;**39**(3):251-260

[12] Wistuba M, Rawat A, Pedapati T. ar Xiv preprint ar Xiv: 1905.01392. 2019

[13] Ojha VK, Abraham A, Snášel V. Metaheuristic design of feedforward neural networks: A review of two decades of research. Engineering Applications of Artificial Intelligence. 2017;**60**:97-116

[14] Lee KY, Cha YT, Park JH. Short-term load forecasting using an artificial neural network. IEEE Transactions on Power Systems. 1992;**7** (1):124-132

[15] Banda E. Department of Electrical Engineering, University of Cape Town, Student thesis. 2006

[16] Liu CL, Yang TY. Study on method of GPS height fitting based on BP artificial neural network. Journal of Southwest Jiaotong University. 2007;**2**(5)

[17] Wu JB, Li WJ. Study on textile industry using BP neural networks. Progress in Text Science and Technology. 2007;**2**(2):7-10

[18] Eberhart RC, Shi Y, Kennedy J. Swarm Intelligence. Morgan Kaufmann Publishers; 2001

[19] Cedeño W, Agrafiotis DK. Using particle swarms for the development of QSAR models based on K-nearest neighbor and kernel regression. Journal of Computer-Aided Molecular Design. 2003;**17**(2):255-263

[20] Lin WQ, Jiang JH, Shen Q, Shen GL, Yu RQ. Optimized block-wise variable combination by particle swarm optimization for partial least squares modeling in quantitative

structure- activity relationship studies. Journal of Chemical Information and Modeling. 2005;**45**(2):486-493

[21] Shen Q, Jiang JH, Jiao CX, Lin WQ, Shen GL, Yu RQ. Hybridized particle swarm algorithm for adaptive structure training of multilayer feed-forward neural network: QSAR studies of bioactivity of organic compounds. Journal of Computational Chemistry. 2004;**25**(14):1726-1735

[22] Yoshida H, Kawata K, Fukuyama Y, Takayama S, Nakanishi Y. A particle swarm optimization for reactive power and voltage control considering voltage security assessment. IEEE Transactions on Power Systems. 2000;**15**(4): 1232-1239

[23] Gudise VG, Venayagamoorthy GK. Comparison of PSO and backpropagation as training algorithms for neural networks

[24] Zhao F, Ren Z, Yu D, Yang Y. Application of an improved particle swarm optimization algorithm for neural network training

[25] Ni HM, Yi Z, Li PC, Tong XF. Application of BP network based on PSO algorithm in cementing quality prediction

[26] Liu C, Ding W, Li Z, Yang C. The International Journal of Advanced Manufacturing Technology. 2017;**89**: 2277-2285

[27] Bai T, Meng H, Yao J. Neural Computing and Applications. 2014;**25**: 1699-1707

[28] Liu T, Yin S. Multimedia Tools and Applications. 2017;**76**:11961-11974

[29] Nandi A, Jana ND. ar Xiv preprint ar Xiv: 1905.04522. 2019

[30] Wang H, Lv Y, Chen H, Li Y, Zhang Y, Lu Z. Multimedia Tools and Applications. 2018;**77**:3871-3885

[31] Zhang Y, Wang S, Sui Y, Yang M, Liu B, Sun J, et al. Journal of Alzheimer's Disease. 2018;**65**:855-869

[32] Le LT, Nguyen H, Dou J, Zhou J, et al. Applied Sciences. 2019;**9**:2630

[33] Chatterjee S, Sarkar S, Hore S, Dey N, Ashour AS, Balas VE. Neural Computing and Applications. 2017;**28**: 2005-2016

[34] Xu Y, Shu Y. Evolutionary Extreme Learning Machine–based on Particle Swarm Optimization. Springer; 2006. pp. 644-652

[35] Pacifico LD, Ludermir TB. Evolutionary extreme learning machine based on particle swarm optimization and clustering strategies. In: 2013 International Joint Conference on Neural Networks (IJCNN) (IEEE). 2013. pp. 1-6

[36] Ling QH, Song YQ, Han F, Lu H. An improved evolutionary random neural networks based on particle swarm optimization and input- to-output sensitivity. In: International Conference on Intelligent Computing. Springer; 2017. pp. 121-127

[37] Van den Bergh F, Engelbrecht AP. IEEE Transactions on Evolutionary Computation. 2004;**8**:225-239

[38] Lin CJ, Chen CH, Lin CT. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 2008;**39**:55-68

[39] Al-Kazemi B, Mohan CK. Training feedforward neural networks using multi-phase particle swarm optimization

[40] Coello CAC, Pulido GT, Lechuga MS. IEEE Transactions on Evolutionary Computation. 2004;**8**: 256-279

[41] Xue Y, Tang T, Liu AX. Large-scale feedforward neural network optimization by a self-adaptive strategy and parameter based particle swarm optimization. IEEE Access. 2019;**7**: 52473-52483

[42] Anand A, Suganthi L. Forecasting of electricity demand by hybrid ann-pso models

[43] Ma Q. Design of BP neural network urban short-term traffic flow prediction software based on improved particle swarm optimization

[44] Xiao Y, Xiao J, Lu F, Wang S. International Journal of Computational Intelligence Systems. 2014;**7**:272-290

[45] Zhang YD, Wang S, Dong Z, Phillip P, Ji G, Yang J. Progress In Electromagnetics Research. 2015;**152**: 41-58

[46] Lu Y, Zeng N, Liu Y, Zhang N. A hybrid wavelet neural network and switching particle swarm optimization algorithm for face direction recognition. Neurocomputing. 2015;**155**:219-224

[47] Chen X, Wang J, Sun D and Liang J 2008 A novel hybrid evolutionary algorithm based on pso and AFSA for feedforward neural network training

[48] Zhang Y, Wu L. Crop classification by forward neural network with adaptive chaotic particle swarm optimization. Sensors. 2011;**11**:4721-4743

[49] Zhang JR, Zhang J, Lok TM, Lyu MR. Applied Mathematics and Computation. 2007;**185**:1026-1037

[50] Yaghini M, Khoshraftar MM, Fallahi M. Hiopga: A new hybrid meta- heuristic algorithm to train feedforward neural networks for prediction

[51] Kartheeswaran S, Durairaj DDC. Informatics in Medicine Unlocked. 2017; **8**:21-31

[52] Zhang C, Shao H. An Ann's Evolved by a New Evolutionary System and Its Application. Vol. 4. IEEE; 2000. pp. 3562-3563

[53] Carvalho M, Ludermir TB. Particle swarm optimization of feed- forward neural networks with weight decay. In: 2006 Sixth International Conference on Hybrid Intelligent Systems (HIS'06). IEEE; 2006. p. 5

[54] Xue B, Ma X, Gu J, Li Y. An Improved Extreme Learning Machine Based on Variable-length Particle Swarm Optimization. IEEE; 2013. pp. 1030-1035

[55] Carvalho M, Ludermir TB. Particle swarm optimization of neural network architectures and weights

[56] Kiranyaz S, Ince T, Yildirim A, Gabbouj M. Neural Networks. 2009;**22**: 1448-1462

[57] Das G, Pattnaik PK, Padhy SK. Expert Systems with Applications. 2014; **41**:3491-3496

[58] Asadnia M, Chua LH, Qin X, Talei A. Journal of Hydrologic Engineering. 2014; **19**:1320-1329

[59] Zhang YD, Wang S, Wu L. Progress in Electromagnetics Research. 2010;**109**: 325-343

[60] Chunkai Z, Yu L, Huihe S. A New Evolved Artificial Neural Network and

Its Application. Vol. vol 2. IEEE; 2000. pp. 1065-1068

[61] Wang H, Wang Y, Wang Y. Expert Systems with Applications. 2013;**40**: 418-428

[62] Qi C, Fourie A, Chen Q. Construction and Building Materials. 2018;**159**:473-478

[63] Yu J, Xi L, Wang S. An improved particle swarm optimization for evolving feedforward artificial neural networks. Neural Processing Letters. 2007;**26**: 217-231

[64] Juang CF. IEEE Transactions on Systems, Man, and Cybernetics (Part B) (Cybernetics). 2004;**34**:997-1006

[65] Meissner M, Schmuker M, Schneider G. BMC Bioinformatics. 2006; **7**:125

[66] Carvalho M, Ludermir TB. An Analysis of PSO Hybrid Algorithms for Feed-forward Neural Networks Training. IEEE; 2006. pp. 6-11

[67] Zhao W. BP Neural Network based on PSO Algorithm for Temperature Characteristics of Gas Nanosensor JCP. Vol. 72012. pp. 2318-2323

[68] Armaghani DJ, Hajihassani M, Mohamad ET, Marto A, Noorani S. Arabian Journal of Geosciences. 2014;**7**: 5383-5396

[69] Pradeepkumar D, Ravi V. Applied Soft Computing. 2017;**58**:35-52

[70] Yang J, Wang L, Jiang Q. Ford vehicle identification via shallow neural network trained by particle swarm optimization

[71] Kong Y, Abdullah S, Schramm D, Omar M, Haris S. Journal of Mechanical Science and Technology. 2019;**33**: 5137-5145

[72] Chen B, Zhang H, Li M. Prediction of pK (a) values of neutral and alkaline drugs with particle swarm optimization algorithm and artificial neural network. Neural Computing and Applications. 2019

Chapter 4

Perspective Chapter: Airborne Pollution (PM2.5) Forecasting Using Long Short-Term Memory Deep Recurrent Neural Network Optimized by Gaussian Process

Marco Antonio Olguin-Sanchez,
Marco Antonio Aceves-Fernández,
Jesus Carlos Pedraza-Ortega and Juan Manuel Ramos-Arreguín

Abstract

Forecasting air pollution is a challenging problem today that requires special attention in large cities since they are home to millions of people who are at risk of respiratory diseases every day. At the same time, there has been exponential growth in the research and application of deep learning, which is useful to treat temporary data such as pollution levels, leaving aside the physical and chemical characteristics of the particles and only focusing on predicting the next levels of contamination. This work seeks to contribute to society by presenting a useful way to optimize recurrent neural networks of the short and long-term memory type through a statistical process (Gaussian processes) for the correct optimization of the processes.

Keywords: deep learning, gaussian process, optimization, recurrent neural network, long-short term memory, airborne pollution

1. Introduction

Recurrent neural networks (RNN), especially Long Short Term Memory (LSTM), have proved their efficiency in working on time-dependent values by (as its names indicate) the use of memory (sequences) gives enough information to the network to work properly finding patterns and trends in the values, which are not so obvious at first glance. Also, the gaussian processes are a useful statistical technique that allows the hyperparameters of the network since it has shown that the processing time is reduced and, at the same time, the accuracy may be improved [1]. Afterward, there is airborne pollution, which is a complex system that affects billions of people worldwide, especially in a metropolis such as Hotan, China. Shanghai, China, Ghaziabad, India, or in the case of this study, Mexico City, Mexico. Also, there we have a lot of types of particles interacting with each other in chemical, biological, and physical ways. The pollutants that are monitored by the SEDEMA's network in the City of Mexico are nitrogen dioxide (NO_2), Ozone (O_3), sulfur dioxide (SO_2) and particulate matter ($PM_{2.5}$, and PM_{10}). Hence, we propose the use of an RNN-LSTM and optimizing its hyperparameters using Gaussian processes to increase the accuracy in the forecast of airborne pollution instead of the use of the current method.

2. Literature review

2.1 Airborne pollution

Air pollution has been a problem that has been increasing in recent decades mainly in large cities, bringing with it respiratory diseases [2, 3]. This contamination is accompanied by the same pollutant particles that have a useful life depending on physical parameters (size and shape) and their chemical composition. Mexico City has been studied for decades [4] due to its high levels of pollution that affect more than 20 million people. The sites used in this work are the following: Northeast (Gustavo A. Madero—GAM, FES Aragón—FAR, Xalostoc—XAL), Northwest (Tlalnepantla—TLA,), Center (Hospital General de México—HGM, Merced—MER), Southeast (Nezahualcóyotl—NEZ, Santiago Acahualtepec—SAC) and Southwest (Ajusco Medio—AJM, Pedregal—PED, Santa Fe, SFE). The map of the monitoring sites is shown in **Figure 1**.

2.1.1 Multiple imputation by chained equations (MICE)

Dealing with the missing data problem often leads to two general approaches for imputing multivariate data: Joint modeling (JM) and fully conditional specification

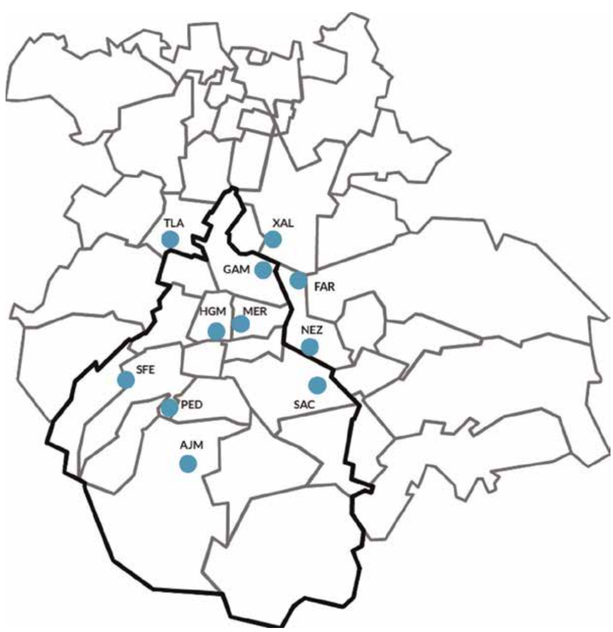


Figure 1.
Location of the monitoring sites in Mexico City.

(FCS), also known as multivariate imputation by chained equations (MICE) [5]. It is known that is a JM-type problem when we must specify a multivariate distribution for the missing data and obtain the imputation of its conditional distributions through Markov Monte Carlo chains (MCMC) techniques. On the other hand, it is FCS, which specifies the multivariate imputation model on a variable-by-variable basis using a set of conditional densities, one for each incomplete variable. The imputation starts by iterating over the conditional densities, usually, a low number of iterations is enough. In order to explain the model, let us use the following notation [5]: Let Y_j with ($j = 1,\cdots$, p) be one of p incomplete variables and $Y =, \cdots, Y_p)$. The observed and missing parts of Y_j are denoted by Y_j^{obs} and Y_j^{mis}, respectively, then $Y^{obs} = \left(Y_1^{obs}, \cdots, Y_p^{obs}\right)$ and $Y^{obs} = \left(Y_1^{mis}, \cdots, Y_p^{mis}\right)$, these are the observed and missing data respectively in Y. The number of imputations is $m \geq 1$. The h-th imputed data set is denoted as $Y^{(h)}$ where $h = 1,\cdots,m$. Now let $Y_{-j} = \left(Y_1, \cdots, Y_{j-1}, Y_{j+1}, \cdots, Y_p\right)$ denote the collection of the $p - 1$ variables in Y except Y_j. Finally, let Q denote the quantity of scientific interest. The mice algorithm has three main steps: imputation, analysis, and pooling. The analysis starts with an incomplete data set Y_{obs}. The second step is to compute Q on each imputed data set, here the model is applied to $Y^{(1)},\cdots,Y^{(m)}$ in the general identical. Finally, the third step is to pool the m estimates $\hat{Q}^{(!)},\cdots,\hat{Q}^{(m)}$ into one estimate $\bar{Q}$ and estimate its variance.

2.1.2 Recurrent neural networks

Recurring neural networks (better known as RNN) can be used for any type of data. In practical applications, the use of symbolic values is more common. In a recurrent neural network, there is a one-to-one correspondence between the layers of the network and specific positions in the sequence. The position in the sequence is also known as its timestamp. Finally, RNNs are complete Turing, which means that this type of network can simulate any algorithm with sufficient data and computational resources [6]. A representation of this kind of network is shown in **Figure 2**.

2.1.3 Long-short term memory (LSTM)

To represent the hidden states of the kth hidden states (layer) the notation $h_t^{-(k)}$ is used and to simplify the notation it will be assumed that the input layer $\bar{x}_t$ can be denoted by $h_t^{-(0)}$ (this layer is not hidden) [7]. To obtain good results, a hidden vector of dimension p must also be included, which will be denoted by $\bar{c}_t^{(k)}$ and refers to the state of the cell. The state of the trap can be observed as the long-term memory within the network. The matrix that updates the values is denoted by $W^{(k)}$ and is used to permute the column vectors $\left[h_t^{-(k-1)}, h_{t-1}^{-(k)}\right]^T$. The matrix that is obtained always results in dimensions $4p \times 2p$. A $2p$ size vector is then premultiplied by the $W^{(k)}$ matrix resulting in a $4p$ vector. Now to find the updates we have the following; for setting up intermediates Eq. 1, for selectively forget and add to long-term memory eq. 2, for selectively leak long-term memory to hidden state (Eq. 3).

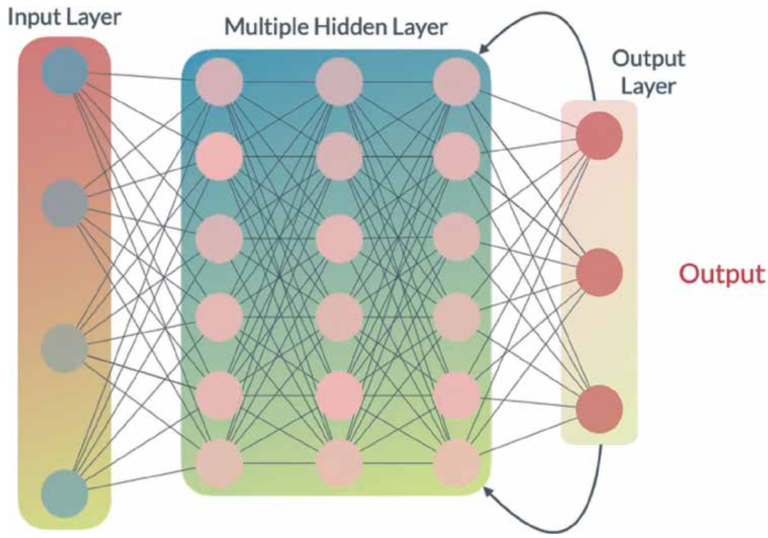


Figure 2.
Graphic representation of a recurrent neural network.

$$\begin{matrix} InputGate: \\ ForgetGate: \\ OutputGate: \\ NewC-State: \end{matrix} \begin{bmatrix} \bar{i} \\ \bar{f} \\ \bar{o} \\ \bar{c} \end{bmatrix} = \begin{pmatrix} sigm \\ sigm \\ sigm \\ sigm \end{pmatrix} W^{(k)} \begin{bmatrix} \bar{h}_t^{(k-1)} \\ \bar{h}_{t-1}^{k} \end{bmatrix} \tag{1}$$

$$c_t^{\overline{(k)}} = \bar{f} \odot c_{t-1}^{\overline{(k)}} + \bar{i} \odot \bar{c} \tag{2}$$

$$h_t^{\overline{(k)}} = \bar{o} \odot tanh\left(c_t^{\overline{(k)}}\right) \tag{3}$$

Additionally, clarify that LSTM is an algorithm that belongs to recurrent neural networks or RNN [8]. The RNN's refer to neural networks that take their previous state as input, this means that the neural network will have two inputs, the new information entered into the network and its previous state, which is shown in **Figure 2**. With this model we can have short-term memory in the neural network [9]. These neural networks have applications in sequential predictions, that is, predictions that depend on a temporal variable.

2.2 Bayesian optimization using Gaussian processes

2.2.1 Multidimensional Gaussian distribution

To talk about Gaussian processes, we must first define a multivariable Gaussian distribution in several dimensions. Formally, this distribution is expressed as in Eq. 4:

$$p(x) = \frac{1}{(2\pi)^{D/2}} e^{\frac{-1}{2}(x-\mu)^T} \Sigma^{-1}(x-\mu) \tag{4}$$

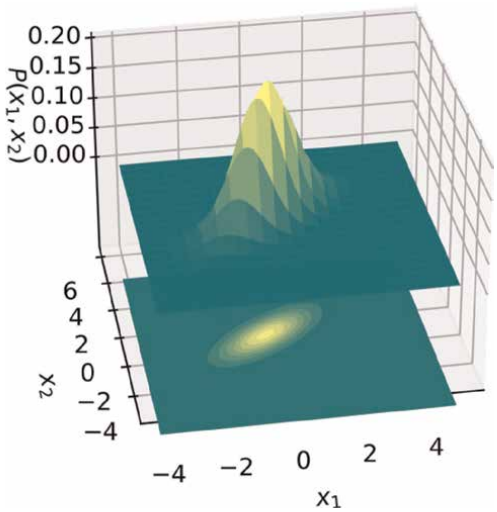


Figure 3.
Graphic example of a multidimensional gaussian distribution.

Where D is the number of dimensions, x is the variables, μ is the average vector, Σ is the covariance matrix. Gaussian processes try to model a function f given a set of points [10]. Traditional nonlinear regression machine learning methods usually give a function that they think best fits these observations. But, there may be more than one function that fits the observations equally well. When we have more observation points, we use our posterior-anterior as our anterior, we use these new observations to update our posterior. This is the Gaussian process. A Gaussian process is a probability distribution over possible functions that fit a set of points. Because we have the probability distribution over all possible functions, we can calculate the means as the function and calculate the variance to show how confident we are when we make predictions using the function (**Figure 3**).

2.2.2 Gaussian process

Because we have the probability distribution over all possible functions, we can calculate the means as the function and calculate the variance to show how confident when predictions are made using the function as demonstrated by Wang [10], we must take into account that:

I. The (later) functions are updated with new observations.

II. The mean calculated by the posterior distribution of the possible functions is the function used for the regression.

The function is modeled by a multivariable Gaussian of the form shown in Eq. 5:

$$P(f|X) = N(f|\mu, K) \tag{5}$$

Where $X = [x_, \cdots, x_n], f =, \cdots, f(x_n), \mu =, \cdots, m(x_n), K_{i,j} = k(x_i, x_j)$. Being X the points of the observed data, m represents the average function and K represents a definite positive kernel.

3. Materials and methods

3.1 Materials

The data used to train the model were obtained from the Atmospheric Monitoring System (SIMAT for its acronym in Spanish) database is conformed of four subsystems: RAMA, REDMA, REDMET, and REDDA, all are given by its website but we just focus on the Automatic Environmental Monitoring Network (RAMA for its acronym in Spanish).

3.2 Methodology

3.2.1 Data acquisition and preprocessing data

First, the database on air quality, RAMA (SEDEMA, 2021) [11] must be downloaded, which is public. This file (dataset) contains values captured by all stations capable of monitoring $PM_{2.5}$. In case the dataset contains more variables in addition to the one already mentioned, a preprocessing process must be carried out. First, the values of interest ($PM_{2.5}$. and air direction) should be classified, excluding any other. Once the data has been classified, it will be necessary to determine if there are missing data and in which cases it is convenient to impute because if the amount of data to be imputed exceeds 40% of the total data, it is advisable not to use that station since there is a loss very large data and a case of over-learning or data that does not reflect reality could be presented. To impute the missing data, the MICE algorithm will be used and once the algorithm has been applied, a new dataset will have to be generated with the imputed data (complete). Because the RNN-LSTM works by taking a tensor as input and already having an absent dataset of missing data, now it will be necessary to divide this data into three different datasets which would serve for training, testing, and data validation. To conclude with the preprocessing, the data will be normalized and later converted into tensors. To normalize the data, the min-max normalization will be used, which takes the minimum value of the data as "zero" and the maximum value as "one" and it is based on these that the normalization is performed. For tensors, they must take into account the batch value, which in turn takes values of 2^n with $n \epsilon N$. The value of the sequence to use as a parameter should also be considered when creating the tensors.

3.2.2 Instantiate LSTM and optimize the model

To begin with the training and optimization of the network, we are going to start the network with values of the hyperparameters selected at random, this is only to make the network start and work since later the values of each selected hyperparameter will be rewritten in optimization until the optimal point is found within the search space that is established. By having the data imputed, divided, normalized, and transformed into tensors now, using the training dataset, as its name implies, we will begin the process of training the network, which will go hand in hand with the number of iterations (points) that have been assigned to the optimization process. In each iteration, an adjustment will be made in some of the hyperparameters, and saving the score of each model to end up with the best one.

3.2.3 Model evaluation

Having already a trained and optimized model, we can now determine the efficiency of the model by calculating the RMSE that the model has by comparing the predicted data against the real data. To generate predictions, we are going to use the evaluation set (**Figure 4**).

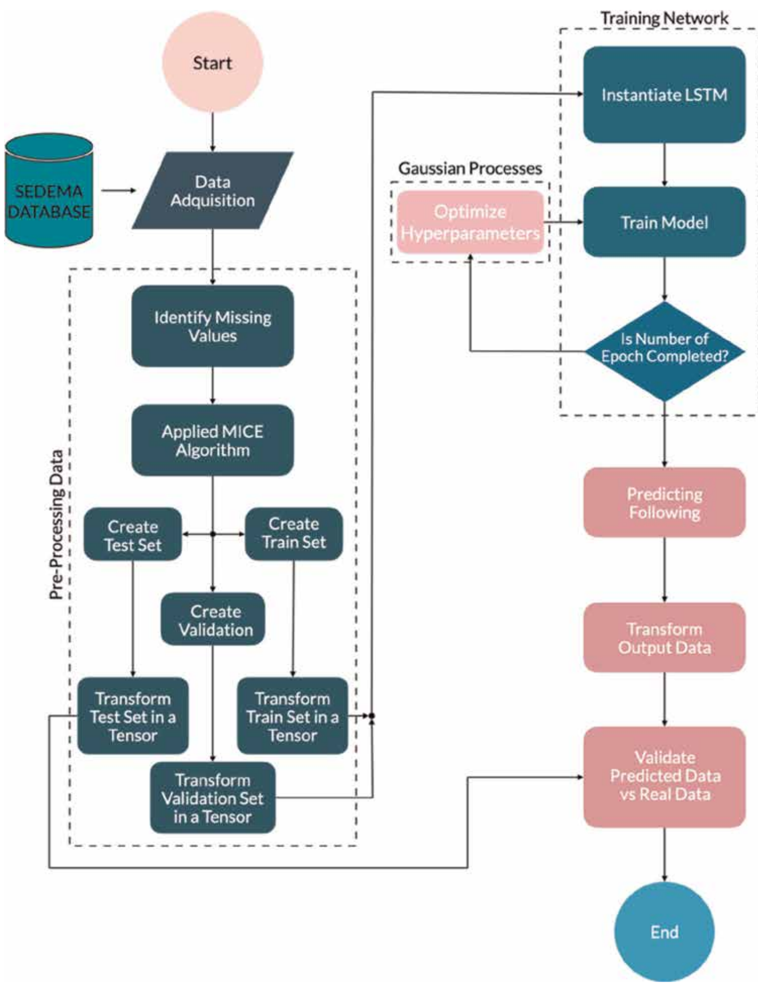


Figure 4.
Proposed methodology.

4. Results and discussion

There was a considerable amount of missing values all across the data. After applying the MICE algorithm the data full fill. In order to confirm that the imputation was successfully checked that the distribution of the data did not change as is shown in **Figures 5** and **6**. These processes will be repeated for each station. This process was repeated for all the stations which were selected getting similar results in all cases. Once the preprocessing was finished, the training was started and by optimizing the model, a search space was stabilized for each hyperparameter of the RNN-LSTM considered. Now, with the model ready and tested, the results of the optimization process can be seen within the network. As is shown in **Figures 7** and **8**, for the LSTM optimized by GP, the loss function during training decreases rapidly in each epoch until it converges and the change between epochs is no longer so noticeable compared with simple LSTM (**Figures 9** and **10**). In both cases when the converges are reached, it means that the network has already stopped learning. Finally, the validation set is used to estimate the skills of the network obtained in its training for the prediction of $PM_{2.5}$ levels. This is shown for the LSTM optimized by GP in **Figures 11** and **12**, and

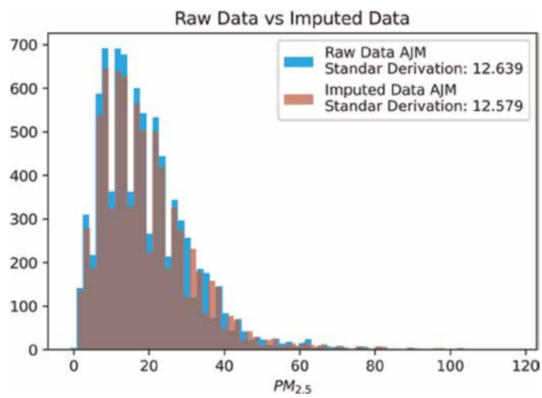


Figure 5.
Data distribution of AJM station before and after being imputed.

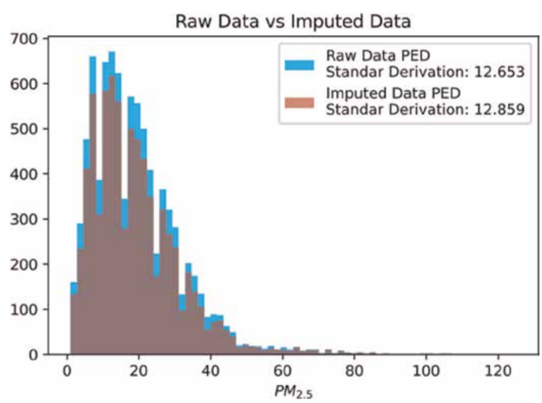


Figure 6.
Data distribution of PED station before and after being imputed.

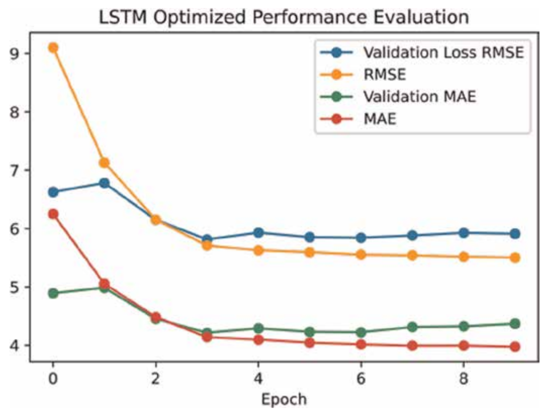


Figure 7.
Metrics obtained with LSTM model optimized by Gaussian process for the AJM station.

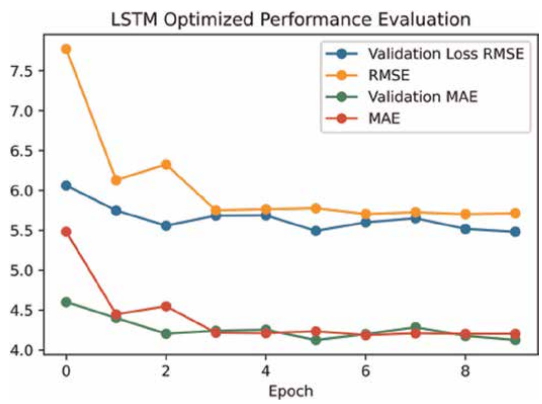


Figure 8.
Metrics obtained with LSTM model optimized by Gaussian process for the PED station.

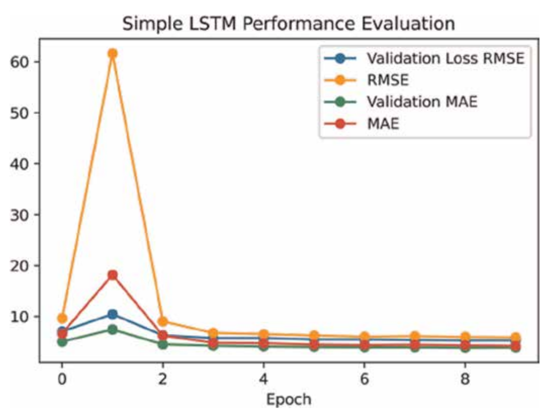


Figure 9.
Metrics obtained with simple LSTM model for the AJM station.

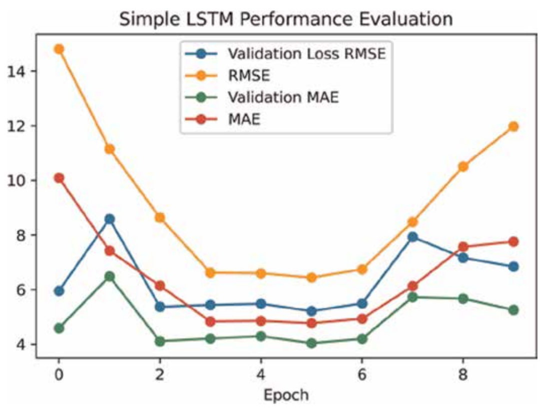


Figure 10
Metrics obtained with simple LSTM model for the PED station.

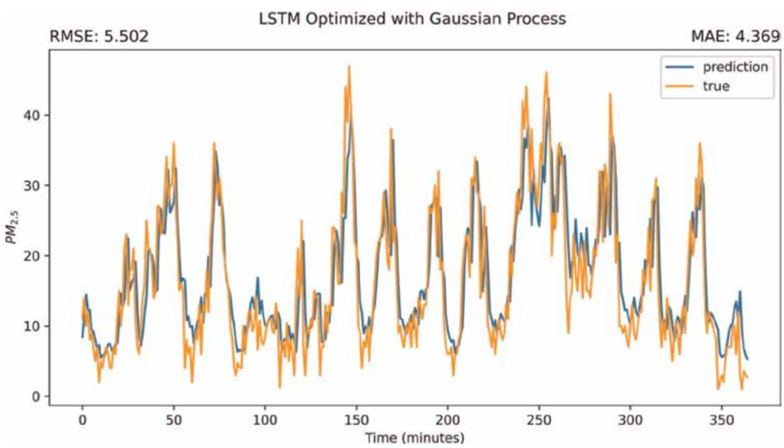


Figure 11.
LSTM optimized by gaussian process, validation forecast for AJM station.

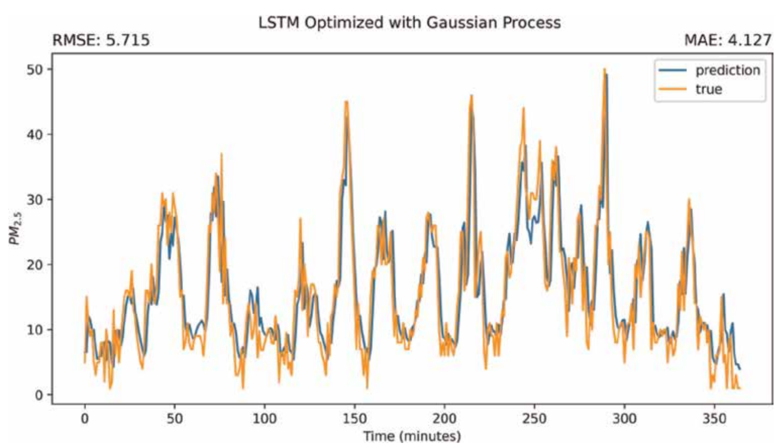


Figure 12.
LSTM optimized by gaussian process, validation forecast for PED station.

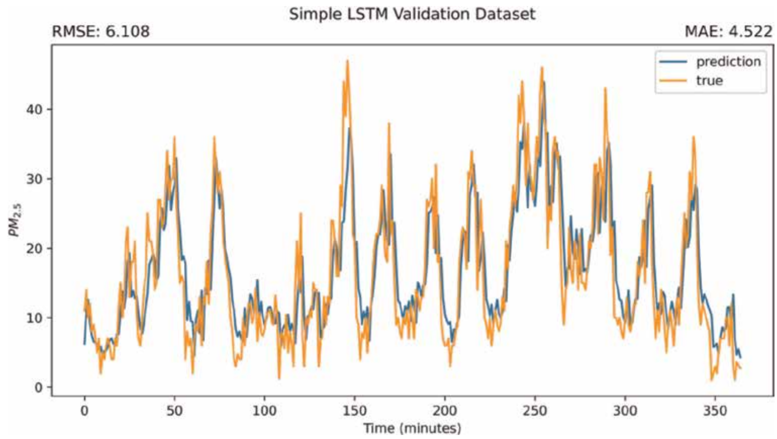


Figure 13.
Simple LSTM, validation forecast for PED station.

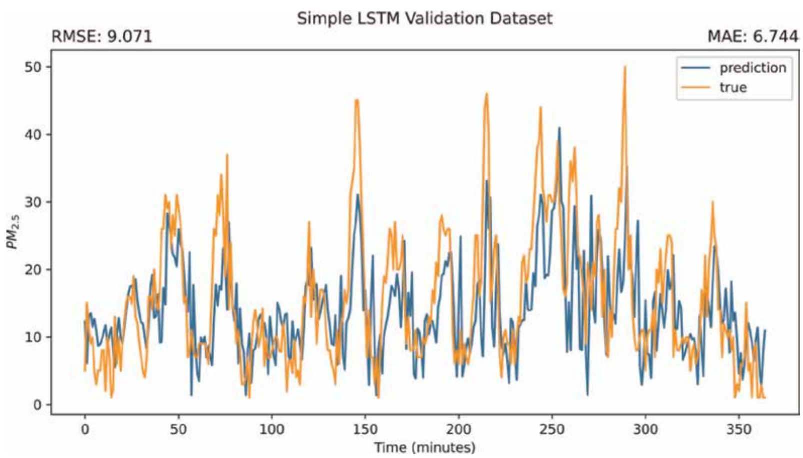


Figure 14.
Simple LSTM, validation forecast for PED station.

for the simple LSTM the results are shown in **Figures 13** and **14**. The results show are from AJM and PED stations.

5. Conclusion

The potential of using Gaussian Processes to improve the hyperparameters tuning is based on a strong mathematical model compared with other methods of hyperparameter tuning; this gives more confidence when looking for an optimized deep learning model. On the other hand, to use this method requires more time to compute the Bayesian search and it may be a thing to consider.

About the project, it does not pretend to end the airborne pollution problem, but it introduces a useful tool for the government and the citizen in order to prevent and plane his day because the model can make predictions along more than three hundred hours with high accuracy, so this gives 12 days to the corresponding users to avoid or prevent the contamination levels.

Conflict of interest

The authors declare no conflict of interest

Consent for publication

Not applicable

Author details

Marco Antonio Olguin-Sanchez, Marco Antonio Aceves-Fernández*,
Jesus Carlos Pedraza-Ortega and Juan Manuel Ramos-Arreguín
Autonomous University of Queretaro, Queretaro, Mexico

*Address all correspondence to: marco.aceves@uaq.mx

References

[1] Sánchez CJ. Características fisicoquímicas de los gases y partículas con- taminantes del aire. Su impacto en el asma. [Internet]. Available from: www.scielo.org.co/scielo.php?script=sci abstract&pid=S0121-07932012000400007&lng=en&nrm=iso&tlng=es [Accessed: October 10, 2022]

[2] Arbex MA, Saldiva PH, Pereira LA, Braga AL. Impact of outdoor biomass air pollution on hypertension hospital admissions. Journal of Epidemiology and Community Health. 2010;**64**(7):573-579. DOI: 10.1136/jech.2009.094342

[3] Guo Y, Barnett AG, Zhang Y, Tong S, Weiwei Y, Pan X. The short-term effect of air pollution on cardiovascular mortality in Tianjin, China: Comparison of time series and case–crossover analyses. Science of The Total Environment. 2010;**409**(2):300-306. DOI: 10.1016/j.scitotenv.2010.10.013

[4] Stephens SM, Sasha W, Olson F, Ramos J, Retama R, Armando, et al. Weekly patterns of México City's surface concentrations of CO, NOx, PM10, and O3 during 1986-2007. Atmospheric Chemistry and Physics. 2008;**8**: 5313-5325. DOI: 10.5194/acpd-8-8357-2008

[5] van Buuren S, Groothuis-Oudshoorn K. Mice: Multivariate imputation by chained equations in R. Journal of Statistical Software. 2011;**45**(3):1-67. DOI: 10.18637/jss.v045.i03

[6] Siegelmann HT, Sontag ED. On the computational power of neural nets. Association for Computing Machinery. 1992;**1995**:440-449. DOI: 10.1145/130385.13043

[7] Aggarwal CC. Neural Networks and Deep Learning. Midtown Manhattan, New York City: Springer Cham; 2018. pp. 293-294. DOI: 10.1007/978-3-319-94463-0

[8] Kuri-Monge GJ, Aceves-Fernández MA, Ramírez-Montañez JA, Pedraza-Ortega JC. Capability of a recurrent deep neural network optimized by swarm intelligence techniques to predict exceedances of airborne pollution (PMx) in largely populated areas. In: 2021 International Conference on Information Technology (ICIT), 2021. Amman, Jordan: IEEE; 2021. pp. 61-68

[9] Ramírez Montañez JA, Aceves Fernandez MA, Arriaga ST, Ramos Arreguin JM, Salini Calderon GA. Evaluation of a recurrent neural network LSTM for the detection of exceedances of particles PM10. In: 2019 16th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE). Mexico City, Mexico: IEEE; 2019. pp. 1-6

[10] Wang J. An Intuitive Tutorial to Gaussian Processes Regression. [Internet]. 2020. Available from: https://www.researchgate.net/publication/344359964_An_Intuitive_Tutorial_to_Gaussian_Processes_Regression [Accessed: February 22, 2022]

[11] SEDEMA. Red automática de monitoreo ambiental [Internet]. 2018. Available from: http://www.aire.cdmx.gob.mx/ [Accessed: February 23, 2021]

Chapter 5

Dynamic Economic Load Dispatch of Hydrothermal System

Soudamini Behera, Ajit Kumar Barisal and Sasmita Behera

Abstract

A Quasi Oppositional Gray Wolf Optimization (QOGWO) algorithm has been used in this work to decipher the economic load dispatch of hydrothermal system. Dynamic economic load dispatch problem involves scheduling of committed generators to meet the load demand with minimum fuel cost and several constraints which are dynamic in nature. It is basically short-term hydrothermal scheduling (STHS) problems through cascaded reservoirs. Instead of pseudo-random numbers quasi-opposite numbers are used to initialize population in the proposed QOGWO method so that the convergence rate of GWO increases. The viability of the projected approach is verified in three standard multi-chain cascaded hydrothermal systems with four interconnected hydro systems. The load and number of thermal units differ from one system to another. Water transportation delay between interconnected reservoirs, Valve Point Loading (VPL) have been considered in different combination in three cases. The technique put forth with established superior to many recent findings for the STHS problems with increased complexities.

Keywords: hydrothermal scheduling, cascaded reservoir, gray wolf optimizer (GWO), quasi oppositional-based learning, STH problem, VPL effect

1. Introduction

Over the last few years, we are in a shortage of energy and facing the environmental pollution problem. So, now a day's wise utilization of energy and the operating cost minimization are major issues in the energy field. This signifies constraints of hydrothermal systems must be modified and more robust technique is required to provide more accurate scheduling results. The main aim of optimal HTS of an electrical system is to optimize hydrothermal generations so that the load demand is fulfilled in a scheduled time with accommodating several system constraints of the hydrothermal system. It is very complicated than that of the thermal system due to nonlinearity.

The Stochastic methods like Genetic Algorithm (GA) [1], Quick Evolutionary Programming (QEP) [2], Improved Particle Swarm Optimization (IPSO) [3], Teaching Learning Based Optimization (TLBO) TLBO [4], Symbiotic Organisms Search (SOS) [5], Intensified water cycle approach (IWCA) [6] have used for solving STH problems. GWO [7] is a simple, fast and effective global optimization method. GWO algorithm has been applied for the solution of non-convex and dynamic economic load

IntechOpen

dispatch problem (ELDP) of electric power system [8]. GWO has successfully solved various ELD problems [9]. Many researchers have demonstrated that an opposite candidate gives a more optimal solution than the candidate. Opposition Based Gray Wolf Optimizer (OGWO) has been implemented in solving ELD problem [10] for thermal power generators which increases the success rate and the convergence speed of GWO.

This study applies Quasi Opposition based GWO (QOGWO) for solving HTS problem of a hydrothermal system which prime objective is to allocate the hydro generation between the multi-reservoir cascaded units with PDZ and thermal units with VPL effect. The objective is to cut the total fuel cost of the thermal system with accommodating several limitations of the hydrothermal system which makes it a non-convex problem. To establish that the intended approach is better, a rigorous exercise of the QOGWO for a hydrothermal system, with the gradual increase of complexity and dimension, is considered in this study. In contrast to recent techniques, the outcomes of the QOGWO technique exhibits superiority for operating cost as well as the convergence characteristics to achieve the optimal result in all the cases tested here.

2. Formulation of STHS Problem

The STHS problem is to allocate the generation to the hydro and thermal units so that the required load demand is achieved and it reduces the net cost without affecting other constraints of Hydro and Thermal systems.

2.1 Hydro-thermal scheduling (HTS)

Since hydropower unit's fuel cost are trivial when assessed with that of thermal unit, the optimal HTS solution lessens the net coal cost of the thermal units with the maximum utilization of the accessible hydro resource. In line with this, the optimal HTS problem is formulated as the fuel cost *FC* as given in (1).

$$FC\left(PT_{i,j}\right) = \sum_{i=1}^{Ns}\sum_{j=1}^{Z} a_i PT_{i,j}^2 + b_i PT_{i,j} + c_i \tag{1}$$

Considering the VPL effect as a sinusoidal variation the Eq. (1) modifies to (2).

$$FC\left(PT_{i,j}\right) = \sum_{i=1}^{Ns}\sum_{j=1}^{Z} a_i PT_{i,j}^2 + b_i PT_{i,j} + c_i + \left|d_i \times \sin\left(e_i \times \left(PT_{i,\quad\min} - PT_{i,j}\right)\right)\right| \tag{2}$$

The prime goal of HTS problem is to reduce the net fuel cost *F* of the thermal plants. Then the objective function is given in (3).

$$Minimize\ F = \sum_{j=1}^{Z}\sum_{i=1}^{N_s} FC\left(PT_{i,j}\right) \tag{3}$$

where the symbols carry the meaning as defined earlier.

The following operational restrictions are to be satisfied.

a. *Load Demand constraints*: It is defined as the balance of the net hydro and thermal generation with the load inclusive of losses in each slot of scheduling j as given in (4)

$$\sum_{i=1}^{N_s} PT_{i,j} + \sum_{i=1}^{N_h} PH_{i,j} = PD_j + PL_j, \text{for } j = 1,2,\dots,Z \tag{4}$$

b. *Generation constraints of Thermal Plant*: The i_{th} thermal generator must operate within the lower and upper bound $PT_{i\,\min}$ and $PT_{i\,\max}$ respectively as shown in (5)

$$PT_{i\,\min} \le PT_{ij} \le PT_{i\,\max} \tag{5}$$

c. *Generation constraints of Hydro Plant*: The i_{th} hydro plant generator must operate between its minimum and maximum bounds $PH_{i\,\min}$ and $PH_{i\,\max}$ respectively as given in (6)

$$PH_{i\,\min} \le PH_{i,j} \le PH_{i\,\max} \tag{6}$$

d. *Reservoir constraint*: The i_{th} reservoir volume capacity has to lie within the lowest and highest margins as expressed in (7)

$$V_{i\,\min} \le V_{i,j} \le V_{i\,\max} \tag{7}$$

e. *Water Discharge constraint*: The flow in m^3,q_{ij}, must be in between the lowest and highest margins as given in (8)

f. *Continuity Equation of Hydraulic Network:* The storage capacity of the reservoir must be in between the lower and higher volume margins as given in (9)

$$V_{i(j+1)} = V_{ij} + \sum_{u=1}^{R_u} \left[q_{u(j-\tau)} + s_{u(j-\tau)} \right] - q_{i(j+1)} - s_{i(j+1)} + r_{i(j+1)} \text{ for } j = 1,2\dots,Z \tag{8}$$

Where τ is the time gap for water transportation to the reservoir i from its upstream reservoir u at time slot j and R_u is the combination of the upstream hydraulic reservoirs before the hydro plant i

g. *The power generation of the hydro plant PH_{ij}*. It depends on water discharge rate and reservoir storage capacity. It is expressed as in (10)

$$PH_{ij} = c_{1i} V_{ij}^2 + c_{2i} q_{ij}^2 + c_{3i} \left(V_{ij} q_{ij} \right) + c_{4i} V_{ij} + c_{5i} q_{ij} + c_{6i} \tag{9}$$

Where, c_{1i} to c_{6i} are the constants.

3. GWO

GWO is a recent soft computing approach that mimics the social activities of gray wolves. This algorithm depicts leadership, tracking, surrounding and attacking prey

[7] activities of the species. In this algorithm a specific number of gray wolves in a group travel through a multi-dimensional search space in search of prey. The position of gray wolves are considered as different position variables and the distances of the prey from the gray wolves determine the fitness value of the objective function. The individual gray wolf adjusts its position and moves to the better position. The GWO saves the best solutions obtained through the course of iterations. The goal of this algorithm is to reach to the prey by the shortest possible route. The movement of each individual is influenced by four processes. Their hunting mechanism is as follows:

a. The initial step of hunting is to track, chase and approach the prey.

b. The second step is to pursue, move around and harass the prey until it gives up.

c. The last step is to attack prey.

These steps are shown in **Figure 1** [7].

The GWO algorithm was anticipated by Mirjalili et al. [7]. Gray wolves are related to the Canidae family and are zenith predator. A pack approximately consists of a group of 5to12 wolves. Their society is divided on the basis of hierarchy. The leader is a couple called the 'Alphas'. They take all the decisions for the pack and these decisions are then communicated to the pack. All the members of the pack respect the leader with keeping their tails down. The alpha is the best member who can manage the pack in a better way. The second level in this hierarchy is the 'Beta' wolves. It is an assistant wolf next to alpha after the current wolf gone. It assists alpha and keeps discipline in the pack. The third level in this hierarchy is the 'Delta' wolves. The lowest ranked gray wolf is 'Omega'. They are the scapegoat or the babysitters. Amidst all the social hierarchy, there is an exciting social activity of the gray wolf is group hunting (optimization).

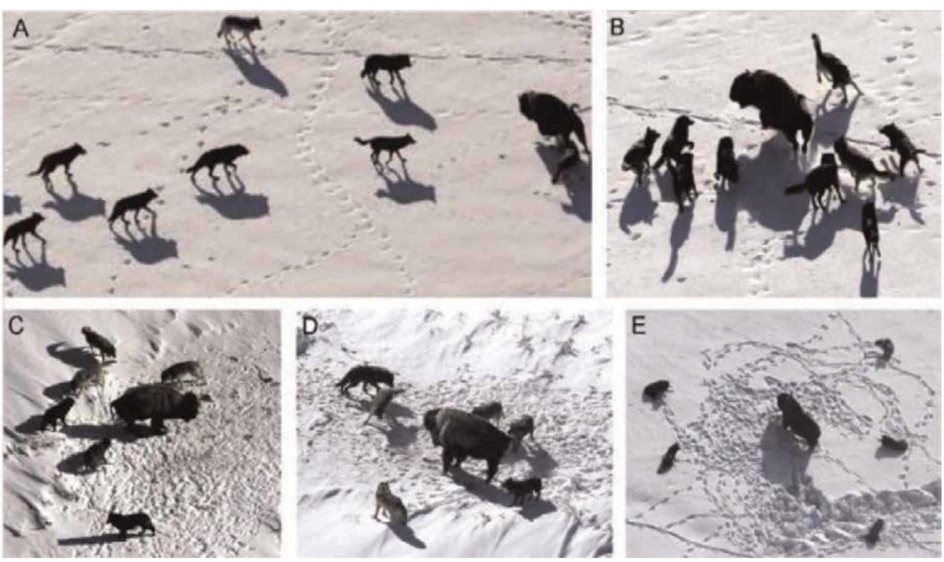


Figure 1.
Hunting steps of Gray Wolf: (A) chasing, approaching and tracking prey (B–D) pursuing, harassing, and encircling (E) Stationary situation and attacking [7].

The encircling behavior of gray wolves may be modeled mathematically as per following Eqs. (11) and (12).

$$\vec{D} = \left|\vec{C}.\vec{X}_p(t) - \vec{X}(t)\right| \tag{10}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A}.\vec{D} \tag{11}$$

Where $\vec{X}_p$ and $\overrightarrow{X}$ are the respective vectors corresponding to the position of the prey and the gray wolf, and t designates the present iteration.

The coefficient vectors $\vec{A}$ and $\vec{C}$ can befound out as given in (13) and (14).

$$\vec{A} = 2\vec{a}.\vec{r}_1 - \vec{a} \tag{12}$$

$$\vec{C} = 2.\vec{r}_2 \tag{13}$$

Where $\overrightarrow{r}_1$ and $\vec{r}_2$ vectors are randomly chosen in the range [0, 1], the values of $\vec{a}$ are gradually varied from 2 to 0 with the increase of iteration so as to put emphasis on exploration and exploitation, respectively.

All the wolves of the pack keep updating their locations as per the location of the senior wolves in the pack. Moreover, the location of the prey would be a random place encircled by the alpha, beta and delta during their search because they are more experienced in hunting. The following Eqs. (15)–(17) are proposed to revise the position of all wolves as per the locations obtained so far by the best candidates as the alpha, beta and delta.

$$\vec{D}_\alpha = \left|\vec{C}_1.\vec{X}_\alpha - \vec{X}\right|, \vec{D}_\beta = \left|\vec{C}_2.\vec{X}_\beta - \vec{X}\right|, \overrightarrow{D}_\delta = \left|\vec{C}_3.\vec{X}_\delta - \vec{X}\right| \tag{14}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.\left(\vec{D}_\alpha\right), \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.\left(\vec{D}_\beta\right), \quad \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.\left(\vec{D}_\delta\right) \tag{15}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{16}$$

Diverging and converging towards the prey in order to search and attack the prey is what the gray wolves follow. In the mathematical modeling of divergence, we use the value of $|A| > 1$ or $|A| < 1$ for the searching wolf to deviate from the prey to emphasize exploration.

At the beginning of search process, a pack of gray wolves is randomly initialized in the GWO algorithm. Each wolf in the searching place updates its gap from the prey. Finally, the algorithm ends the optimization when termination limit is attained.

4. Quasi opposition based learning

Opposition Based Learning considers both the current and its opposite population simultaneously for getting the best candidate solution. The quasi-opposite population $QOP\left(x_1^{q0}, x_2^{q0}, \ldots\ldots\ldots x_i^{q0}, \ldots\ldots x_d^{q0}\right)$ in D dimensional region differs from the opposite population as it is the population between the Centre$c$ of the search region and the opposite point x_i^0, expressed as in (18).

$$x_i^{q0} = rand\left(\frac{a_i + b_i}{2}, a_i + b_i - x_i\right) = rand\left(c_i, x_i^0\right); i = 1,2,.\dots d \tag{17}$$

Where, x_i^{q0} is an arbitrary number between c_i and x_i^0.

4.1 QOGWO algorithm for hydrothermal scheduling

The flowchart of the QOGWO to clarify the HTS problem is given in **Figure 2** and the steps are described as follows:

Step 1: Specify the system parameters, the highest and lowest limits of each variable such as $Pop_{\max}$, $q_{\min}$, $q_{\max}$, N_s, N_h, Z, B-coefficient matrix, P_D, $PT_{\min}$, $PT_{\max}$, $PH_{\min}$, $PH_{\max}$, $V_{\min}$, $V_{\max}$, j_r, Cost coefficients and $iter_{\max}$.

Step-2: Initialize randomly the search agents (Gray wolves) among the population and those agents are possible solutions who satisfy the specified constraints.

Step-3: Compute the trial vector (current search agents) $Q_{i,j,k} = [P_1 \ P_2.\dots P_{POP\max}]$ of the population. The random search agent matrix (P_k) is as in (19).

$$P_k = \begin{bmatrix} q_{11} & q_{12} & \cdots & \cdots & q_{1,Nh} \\ q_{21} & q_{22} & \cdots & \cdots & q_{2,Nh} \\ \cdots & \cdots & \cdots & \cdots & \cdot \\ \cdots & \cdots & q_{ij} & \cdots & q_{i,Nh} \\ q_{Z,1} & \cdots & q_{z,j} & \cdots & q_{Z,Nh} \end{bmatrix} \tag{18}$$

Step 4: The discharge rate $q_{i,d}$ of all the reservoirs for all the slots is taken at random within the bounds and repeatedly adjusted after the check to satisfy the first and last reservoir storage volume which is calculated using (20).

$$q_{i,d} = V_{i,1} - V_{i,25} - \sum_{\substack{j-1 \\ j \neq d}}^{Z} q_{i,j} + \sum_{j-1}^{Z} r_{i,j} + \sum_{u=1}^{R_u} \sum_{j=1}^{Z} q_{u(j-\tau)} \tag{19}$$

Step 5: The volume of each reservoir is calculated by the Eq. (9), and then the hydro generation is scheduled over 24 slots by the Eq. (10).

Step 6: Calculate the thermal power at j_{th} slot using load balance Eq. (4). To satisfy the equation $PT(d, j)$ is taken at random and adjusted using Eq. (21) until it does not defy the limits.

$$B_{dd}PT^2(d, j) + \left(2\sum_{i=1}^{Nh+Ns-1} B_{d,i}.PT(i,\ j) - 1\right)PT(d, j)$$
$$+\left(P_D(j) + \sum_{i=1}^{Nh+Ns-1}\sum_{k=1}^{Nh+Ns-1} PT(i,\ j)B_{i,k}PT(k,\ j) - \sum_{i=1}^{Nh} PH(i,\ j) - \sum_{\substack{i=1 \\ i\neq d}}^{Ns} PT(i,\ j)\right) = 0 \tag{20}$$

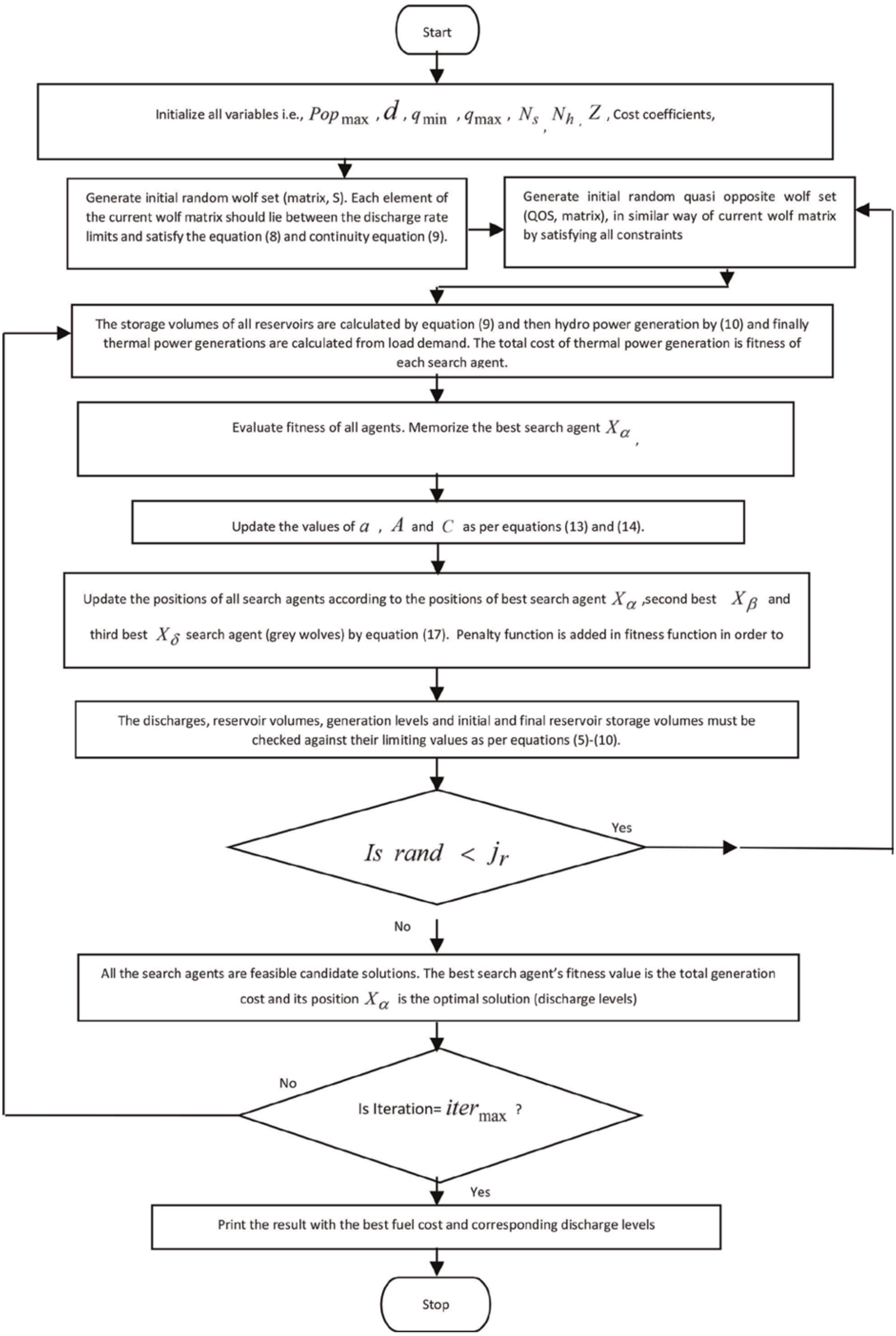


Figure 2.
Flow chart of QOGWO method.

Step 7: The fitness of the solution is evaluated by Eq. (3).
Step 8: Remember the best three search agents X_α, X_β and X_δ (gray wolves) from the population.

Step 9: Revise the values of a, A and C as per Eqs. (13) and (14).
Step 10: According to the positions of best three search agent X_α, X_β and X_δ (gray wolves) all search agents updated their current position by Eq. (17). The violation of constraint is formulated as a penalty added in the fitness function.
Step 11: If the water discharge, the volume of the reservoir and power generations limits are lower than the lowest limit it is assigned the lowest value and if their value exceeds the highest limit, it is assigned the highest value.
Step 12: Select a fresh parameter "jumping rate" (J_r) within the range [0, 1].
If rand $< J_r$, Quasi Opposite set of agents (wolves) can be shaped as below.

$$\begin{gathered} If\ rand < J_{r.} \\ for\ k = 1 : Pop_{max} \\ for\ i = 1 : Z \\ for\ j = 1 : Nh \\ QOS(:\,,\ :\,, k) = q(i,j) = rand(c(j), x_0(j)); \\ end \\ end \\ end \end{gathered} \tag{21}$$

Step 13: Go to Step 2 until the predefined highest iteration number is reached.

5. Result discussion

The projected QOGWO has been used to find the solution of a hydrothermal test system. It has been simulated using MATLAB software. As HTS is a real time problem so, it is necessary that each run of the program should reach close to optimum solution. 20 independent runs are executed to get the optimum solutions for all the algorithms considered here.

5.1 Test system-1

Here the test system-1 is similar to that in [1] but the supplementary data for VPL effect and PDZ of turbines are taken from [2]. Then the fuel cost of the corresponding thermal unit with VPL is given in (22)

$$\begin{aligned} \mathrm{FC}\left(PT_{\mathrm{i,j}}\right) = \sum_{i=1}^{Ns}\sum_{j=1}^{Z} 0.002PT_{i,j}^2 + 19.2PT_{i,j} + 5000 \\ + \left|700 \times \sin\left(0.085 \times \left(PT_{i,\quad \min} - PT_{i,j}\right)\right)\right| \end{aligned} \tag{22}$$

The respective minimum and maximum thermal generations correspond to 500 and 2500 MW. The water loss in the spillway and the energy loss in catering the load from the hydro plant are ignored. The respective lowest and highest hydro generation correspond to 0 and 500 MW.

Three cases of the test system-1 such as Case 1 (HTS problem considering quadratic cost function only), Case 2 (with PDZ) and Case 3 (with VPL and PDZ) are under study. The several controlling parameters like a, A, C, size of the pack and maximum iteration number have been tried in this algorithm. The values of a, A, C are varied as per Eqs. (13) and (14), the size of the pack is 30 and the maximum iterations took is 500.

5.1.1 Case 1: (HTS problems considering quadratic cost function only)

This is the simplest case where the PDZ of the hydro units and the VPL effect of the steam power plant are neglected. The convergence characteristic in **Figure 3** gives an idea about the working of projected QOGWO approach. From **Figure 3** it is clear that the fuel cost is reduced in 50 numbers of iterations. The considered QOGWO approach takes the computation time of 340.452 s to get the optimal HTS. To validate the proposed QOGWO method, its simulation outcomes are compared in terms of best, average and worst fuel cost over 20 independent runs with the results of other approaches as shown in **Table 1**. The optimal results found by the projected algorithm

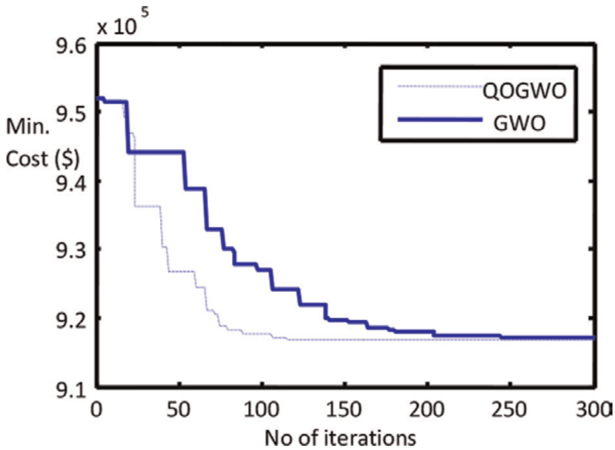


Figure 3.
Convergence features of QOGWO in case-1 of the test system.

Algorithm	Best fuel cost ($)	Average fuel cost ($/day)	Worst fuel cost ($/day)	Variance	Standard deviation	CPU time (sec)
GA [1]	932734.00	936969.00	939734.00	—	—	—
FEP [2]	930267.92	930897.44	931396.81	—	—	—
CEP [2]	930166.25	930373.23	930927.01	—	—	—
IFEP [2]	930129.82	930290.13	930881.92	—	—	1033.20
IPSO [3]	922553.49	—	—	—	—	—
TLBO [4]	922373.39	922462.24	922873.81	—	—	—
SOS [5]	922332.17	922338.20	922482.90	—	—	6.21
GWO	917203.73	917242.58	917288.03	0.0127	0.1128	353.224
QOGWO	916795.74	916812.67	916829.28	0.0096	0.0982	340.452

Table 1.
Comparison of optimal costs for the test system (case 1) after 20 independent runs.

are contrasted with other referred results shown in **Table 1**. It is clear that the QOGWO founded superior result than the above-mentioned accessible techniques. Though SOS has taken less time with a smaller number of iterations and population size, it gives the minimum cost but its minimum is higher than that by GWO and QOGWO.

5.1.2 Case 2: (with PDZ)

The PDZs of reservoirs of hydro power units have taken into account to ensure the viability of the projected method. This case has not been dealt with by many researchers but it is an important case for operation. The results of the proposed method QOGWO are compared in terms of best, average and worst fuel cost over 20 independent runs with the results of other approaches as shown in **Table 2**. It is observed that the QOGWO decreased the minimum, average and worst costs at less execution time than those obtained by the other existing techniques when population size and iterations are similar. The cost convergence feature of QOGWO algorithm is revealed in **Figure 4**.

5.1.3 Case 3: (with VPL and PDZ)

Now the VPL of thermal power units and PDZ of hydro power units are included to confirm the robustness of the projected algorithm. The best rates of hydro

Algorithm	Best fuel cost (\$/day)	Average fuel cost (\$/day)	Worst fuel cost (\$/day)	Variance	Standard deviation	CPU time (s)
IPSO [3]	923443.17	—	—	—	—	—
TLBO [4]	923041.91	—	—	—	—	—
SOS [5]	922844.78	922867.24	923125.44	—	—	9.53
GWO	923146.941	923187.45	923239.50	0.0129	0.1138	321.47
QOGWO	922736.233	922764.186	922810.58	0.0087	0.0932	310.941

Table 2.
Comparison of optimal costs in case-2 for the test system (case 2) after 20 independent runs.

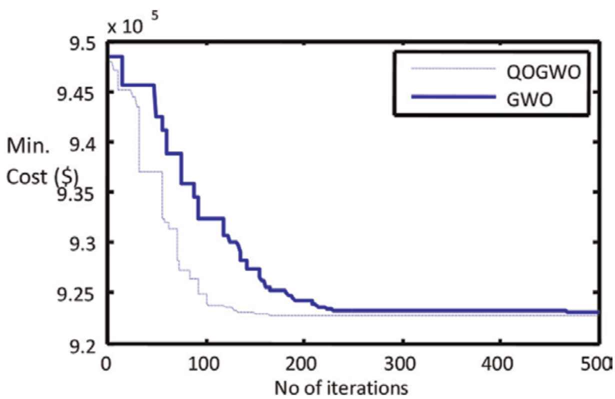


Figure 4.
Convergence features of QOGWO in case-2 of the test system.

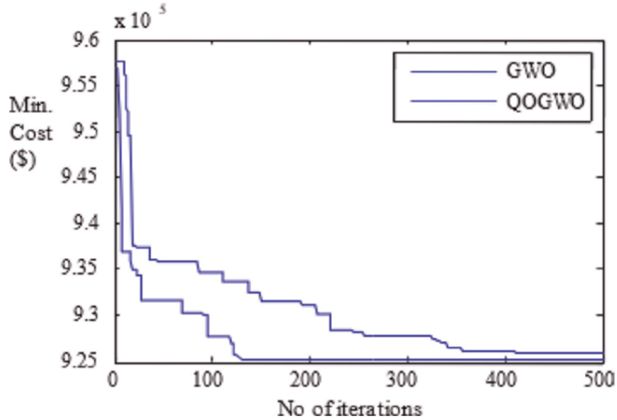


Figure 5(a).
Convergence features of QOGWO in Case-3 of the test system.

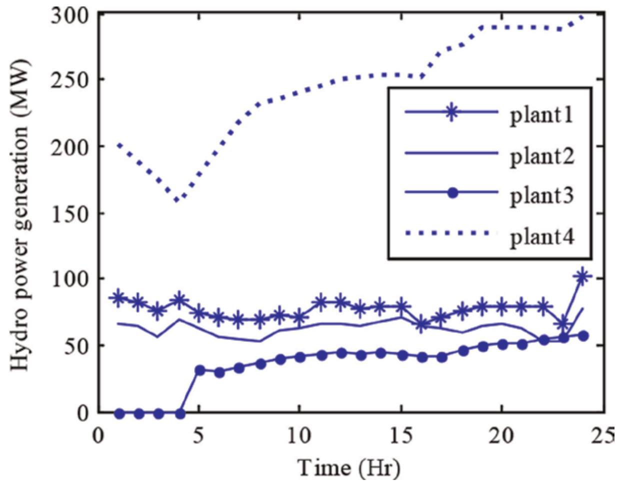


Figure 5(b).
Hourly hydro power generation obtained by QOGWO in Case-3 of the system.

discharges in slots got by the projected QOGWO are shown in **Table 3**. The convergence plot attained by QOGWO is illustrated in **Figure 5(a)**, respectively. In this case, the hourly hydropower generations found by the QOGWO method are given in **Figure 5(b)**.

6. Conclusion

In this study, an effective GWO algorithm is united with quasi-oppositional based learning (QOGWO) has been effectively implemented to solve a hydrothermal test system with quadratic nonlinear cost functions. Progressive improvement of the computational efficiency and better convergence characteristics are attained by quasi-oppositional based learning introduced in the conventional GWO algorithm. It is observed that the simulation time for the same number of iterations and the net cost of generation got by the presented QOGWO for the day is lower than others in all the

Time	Q1	Q2	Q3	Q4	PT (MW)	Time	Q1	Q2	Q3	Q4	PT (MW)
1	9.8564	8.9223	29.7264	13.0645	1017.4263	13	7.9960	8.2150	15.5132	13.4658	1793.6003
2	9.4124	8.5691	29.7217	13.0808	1054.3506	14	7.9872	8.9049	15.9866	13.5550	1756.6392
3	7.9999	6.9965	29.4505	13.1635	1054.1838	15	7.9941	9.5532	16.6901	13.6818	1682.7095
4	9.7886	9.1209	28.6931	13.1236	980.4384	16	6.2264	8.5121	17.5552	13.4527	1645.7608
5	7.9964	8.0000	17.1239	13.1762	943.5170	17	6.7799	8.3072	17.3486	15.5098	1682.7143
6	7.6541	7.0000	17.8867	13.0704	1054.3930	18	7.4033	8.0123	16.0462	15.9873	1682.7161
7	7.3230	6.8335	17.3717	13.1604	1276.1570	19	7.9990	9.0929	14.6481	18.0022	1756.5899
8	7.1700	6.6590	16.3193	13.0902	1608.7928	20	7.9991	9.7041	13.7816	18.0023	1793.5605
9	7.5358	8.0000	15.1289	13.0824	1830.5059	21	7.9973	9.1118	10.0840	18.0004	1756.6036
10	7.2597	8.0000	14.5053	13.1423	1904.4793	22	7.9455	6.9873	10.0585	18.3834	1645.7585
11	9.0105	8.4733	13.5320	13.1777	1793.5797	23	6.1837	6.9939	10.3479	18.9757	1387.0315
12	9.0073	8.4597	13.8286	13.4527	1867.5039	24	12.4744	13.5710	11.7416	22.8566	1054.3990

Table 3.
Hourly hydro discharges ($\times 10^4 m^3$) *of the test system in case-3.*

systems with the different level of complexities because of well-balanced exploration and exploitation of the QOGWO algorithm. The maximum cases of hydrothermal scheduling studied here in comparison to the existing works can be referred by researchers in future. The consistent performance of QOGWO in the large dimension of the problem with multiple constraints exposes its potential for application in other engineering domains for constrained nonlinear non-convex engineering optimization.

Author details

Soudamini Behera[1*], Ajit Kumar Barisal[2] and Sasmita Behera[3]

1 Department of Electrical Engineering, Government College of Engineering, Kalahandi, Odisha, India

2 Department of Electrical Engineering, Odisha University of Technology and Research, Bhubaneswar, Odisha, India

3 Department of Electrical Engineering, Veer Surendra Sai University of Technology, Burla, Odisha, India

*Address all correspondence to: soudamini_behera@yahoo.co.in

References

[1] Orero SO, Irving MR. A genetic algorithm modeling framework and solution technique for short term optimal hydro thermal scheduling. IEEE Transactions on Power Systems. 1998;**13** (2):501-518. DOI: 10.1109/59.667375

[2] Sinha N, Chakrabarti R, Chattopadhyay PK. Fast evolutionary technique for short term hydrothermal scheduling. IEEE Transactions on Power Systems. 2003;**18**(1):214-222. DOI: 10.1109/TPWRS.2002.807053

[3] Hota PK, Barisal AK, Chakrabarti R. An improved PSO technique for short term optimal hydrothermal scheduling. Electric Power Systems Research. 2009; **79**(7):1047-1053. DOI: 10.1016/j.epsr.2009.01.001

[4] Roy PK. Teaching learning-based optimization for short term hydro thermal scheduling problem considering valve point effect and prohibited discharge constraint. International Journal of Electrical Power & Energy Systems. 2013;**53**:10-19. DOI: 10.1016/j.ijepes.2013.03.024

[5] Das S, Bhattacharya A. Symbiotic organisms search algorithm for short-term hydrothermal scheduling. Ain Shams Engineering Journal. 2018;**9**(4): 499-516. DOI: 10.1016/j.asej.2016.04.002

[6] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. International Journal of Advances in Engineering software. 2014; **69**:46-61. DOI: 10.1016/j.advengsoft.2013.12.007

[7] Pradhan M, Roy PK, Pal T. Oppositional based grey wolf optimization algorithm for economic dispatch problem of power system. Ain Shams Engineering Journal. 2018;**9**(4): 2015-2025. DOI10.1016/j.asej.2016.08.023

[8] Kamboj VK, Bath, SK, Dhillon JS. Solution of non-convex economic load dispatch problem using Grey Wolf Optimizer. Neural computing and Applications. 2016;**27**(5):1301-1316. DOI: 10.1007/s00521-015-1934-8

[9] Pradhan M, Roy PK, Pal T. Grey wolf optimization applied to economic load dispatch problems. International Journal of Electrical Power & Energy Systems. 2016;**83**:325-334. DOI: 10.1016/j.ijepes.2016.04.034

[10] Kumar A, Dhillon JS. Environmentally sound short-term hydrothermal generation scheduling using intensified water cycle approach. Applied Soft Computing. 2022;**127**: 109327. DOI: 10.1016/j.asoc.2022.109327

Chapter 6

Using Many Objective Bat Algorithms for Solving Many Objective Nonlinear Functions

Iraq T. Abbas and Saja Ayad

Abstract

In this paper, we have relied on the dominant control system as an important tool in building the group of leaders because it allows leaders to contain less dense areas, avoid local areas, and produce a more compact and diverse Pareto front. Nine standard nonlinear functions yielded this result. MaBAT/R2 appears to be more efficient than MOEAD, NSGAII, MPSOD, and SPEA2. MATLAB was used to generate all the results of the proposed method and other methods in the same field of work.

Keywords: Many-objective problems, bat algorithm, inverted generational distance

1. Introduction

Although the truth of the algorithmic strategy for dealing with combinatorial optimization (CO) has been available for a long time, further application of evolutionary algorithms (EAs) to solve these problems provides a means to deal with large-scale multi-objective optimization.

In this section, the current of my study, which is considered one of the most important studies in recent decades, has been dealt with, and we will explain in it: research objectives, research question, study significance, research breadth, and research limitations.

Often there is not one perfect solution in multi-objective function optimization, but rather a set of optimal Pareto options. Thus, cluster sampling is critical when the co-optimization of an algorithm to generate a comprehensive and varied approximation of the Pareto front (PF) is performed [1]. Using the rule of change of weights, a multi-objective bat algorithm (MOBAT) is introduced to determine the optimal Pareto array for multipurpose functions (MO).

The source [2] also presented bat for multi-objective problem-solving, as well as the multi-objective bat algorithm (MOBAT). To verify this, we will develop solutions against a subset of the multi-objective test functions first. We will now use it to address engineering design improvement challenges such as the total and partial steel beam.

IntechOpen

MOBAT was used for this purpose, and it can be described as a successfully biologically inspired algorithm to address problem floor planning in VSLI design in a publication approach [3].

The author in [4] proposed a multi-purpose optimization problem (MOOP) to achieve both of the aforementioned goals. MOOP is solved using a new simple optimization algorithm called BAT Algorithm, which is based on weight addition method (WSM). Therefore, from the literature we can say here that there is no study before that combined many-objective bat algorithm with indicator convergence R2 (MaBAT/R2). In addition, in another study, a comparison was made between the algorithms for feeding frontal neural networks (NN) and then the gradient descent (GD) algorithms (backpropagation and Levenberg–Marquardt), and three population-based statistical inference methods were used: the bat algorithm, the genetic algorithm (GA), and the particle swarm optimization (PSO) algorithm for the test. It has been shown that the BAT algorithm is superior to all other algorithms in training to feed-forward neural networks (NN) [5]. These results support the use of the best available techniques for further experiments, which greatly contributed to finding the optimal solution.

The advantage of using the bat algorithm is that it allows us to find solutions using population and local search techniques. This work introduced global diversity and rigorous local extraction, both of which are important for exploratory methods. As a result, the Bat algorithm was combined with PSO and local search, in addition to controlling the pulse rate and loudness [6].

MOBAT was used in many-objective optimization problems (MaOPs), which gave us a good balance between diversity and convergence, representing the main issue in MaOPs, by adapting the reference groups approach. Additionally, in 2021, a paper was published entitled using the multipurpose bat algorithm to solve the multipurpose nonlinear programming problem [7]. Moreover, in 2020 [8], a met heuristic hybrid method is proposed to solve multi-objective optimization problems.

We conclude from the above that the main objective of this study is to improve the performance of multi-objective algorithms by developing a new algorithm inspired by bats for multi-objective optimization problems that used a technique to achieve organization and to achieve goals and diversity. Therefore, we proposed a method of increment based on the R2 index distance algorithm to reduce processing efforts in the field of different objective challenges in this paper.

2. Basic concept of optimization problem

In this field, we will first address the general form of the issue of multi-objective optimization and a sequence of definitions and important issues related to the core of the subject under study. So the general form of the problem is:

$$\text{(Minimize) } F(x) = [f_1(x), f_2(x), \ldots, f_k(x)]$$

Subject to:

$$w_i(x) \le 0, i = 1, \ldots, k; \tag{1}$$

$$n_j(x) = 0, j = 1, \ldots, p;$$

$$x_l \ge 0, l = 1, 2, \ldots n$$

where $x = [x_1 : x_2, \ \dots, x_n]^T$ is the vector of decision variables $F_i : R^n \to R; i = 1, \dots, k$ are the objective functions and $wi, n_j : R^n \to R, i = 1,.., m, and\, j = 1, \dots, p$, are constraints functions a problem. To describe the objective concept of optimization, we will give some of the following definitions:

Definition (1) [9]: (Multi-objective optimization problem (MOP)). A MOP is made up of a number of parameters (decision variables), a number of optimization techniques (m), and a number of constraints (m). The determination variables' functions and constraints are functions of the optimization algorithms and requirements. The purpose of optimization is to:

$$Minimize\, y_i = f\ (x_i)\ subj. : toe(x) = (e_1(x); e_2(x); \ \dots; e_k(x)) \leq 0 \tag{2}$$

where $X = (x_1, x_2, x_n)$ and $Y = (y_1, y_2, y_m)$, and x the choice pattern is called the decision vector, the ambition velocity is called the objective vector, the determination space is called the decision sector, and the object space is called the subjective space. The constraints $e(x) \leq 0$ determine the set of feasible solutions.

Definition (2) [9]: (Allocative efficiency optimality). A dimension of choice $x \in X_f$ when it comes to a set, it is said to be completely non $\subseteq X_f iff \nexists a \in A : a \succ x$. If it is evident from the circumstances whichever set A is wanted, the following will simply be omitted. Furthermore, x is described as allocative efficiency optimal *iffx* is nondominated regarding X_f.

Definition (3) [9]: A set of controller parameters in a scalar $x_1 \epsilon X \subset R^n$ is nondominant when it comes to X, if no $x_2 \in X$ appears in the sense that $f(x_2) < f(x_1)$.

Definition (4) [9]: The allocative efficiency optimal set P* is characterized as follows: P* = $\{x_1 \epsilon F : x_1$ is alocative efficiency optimal$\}$.

3. Using bat algorithm to solve MOP

In this section, we will present the new or improved algorithm based on the characteristic of R2 or based on the influencer R2 that was used well and correctly to choose the optimal value when choosing a leader.

Bats are winged mammals and are known to be able to use echolocation. Approximately 996 unique species of bats have been identified worldwide, representing about 20% of all well-evolved mammal species [7]. Another improved computation called BAT [10] is based on the swarm concept. Using BAT, one can re-enact some echolocation features of a smaller level bat. The benefits of this approach include ease of use, versatility, and simplicity in implementation. Moreover, the approach effectively deals with a wide range of challenges, such as highly nonlinear issues. Also, BAT provides a perfect arrangement that promises quickly and works brilliantly with complex problems. Attempting to follow-up are some of the drawbacks of this estimation: conjugation occurs rapidly at first, and the rate of conjugation declines. Furthermore, no scientific study has linked factors to varying rates.

The swarm is responsible for maintaining and re-establishing the perfect Pareto arrangements that have so far been discovered, and which cannot be controlled. The most reasonable arrangement obtained is used in calculating MaBAT/R2. This approach leads people to move in order to find a solution near the best arrangement. Contrasting with Pareto's best suggestions, however, it could not be more objective about space. The Pioneer Choice component is designed to address the research problem under study. The nondominant and most logical arrangements are recorded in a single volume. The leader selects a piece from among the stacked parts of the space layout and suggests one of the nondominant options. The random wheel is used to make the appropriate decision, along with the opportunities available to each individual: Below are full details of the proposed algorithm construction step by step based on the R2 optimum value selection component.

The performance measures in this paper are known as hypervolume (HV) [11] and inverted generational distance (IGD) [12]. Both HV and IGD are able to reflect the focus and diversity of the optimal result set of the algorithms.

Greater similarity to the original PF was indicated by a larger HV value or a smaller IGD number. For many issues, a reference point dominated by true PF is carefully selected to determine HV.

MaBAT/R2 Follow the Steps With R2 Indicator

Set k := 0 and velocity =0 μ=0.1, r0 = 0.5, A = 0.6.
Reload Point at arbitrarily. P_i for n. population ;
Determine the starting Nation's model parameters: f (P);
Discover non-dominated options and use them to start the storage.
WHILE (The requirements for withdrawal have not been met)
BAT Steps
$Q = Qmin + (Qmin - Qmax) * rand$ (**fitness function for bat algorithm**)
$P_{leader1}$ = Choose a leader(archive)
$V_{(t+1)} = V_{(t)} + \left(P_{leader1} - P_{(t)}\right) * Q$ (**velocity function for bat algorithm**)
$P_{new} = P_{(t)} + V_{(t+1)}$ (**position function for bat algorithm**)
If rand > r
$P_{leader2}$ = Choose a leader (archive)(**chosse the ideal value based on R2**)
$P_{new} = P_{(t)} + rand * \left(P_{leader2} - P_{(t)}\right)$
End
if P_{new} *dominated on* $P_{(t)}$ & $(rand < A)$
$P_{(t)} = P_{new}$
End
If rand $<\left(\frac{1-(k-1)}{Max\ iteration-1}\right)^{1/\mu}$
S =Mutation($P_{(t)}$)
if P_{new} *dominated on* $P_{(t)}$ & $(rand < A)$
$P_{(t)}$=S
End
End
Look for options that aren't dominating.
Update the archive with the non-dominated alternatives that have been found.
If the archive is full
To omit one of the current archive members, use the R2 technique.
Make a note of the new solution in the database.
end if
How many of the new archived responses is outside, update the R2 to include the creative approach (s)
end if
increase* r *and reduce A
Set k := k + 1;
End While

4. Experimental results

Now we will present the most important results, which proved the superiority of the proposed algorithm MaBAT/R2 over other algorithms using the well-known functions DTLZ (the DTLZ suite of benchmark problems, created by [13], is unlike the majority of multi-objective test problems in that the problems are scalable to any number of objectives), from which we took only nine functions for comparison and with different sizes in terms of directions, number of target functions, and number of repetitions, especially regarding the problems of irregular Pareto Front (PF) patterns.

4.1 Inverted generational distance (IGD)

Let S denote the search result of a MOEA on a specific MOP. Should R be a set of PF representation points that are equally spaced? [1] Can be used to determine S's IGD value in relation to R.

$$IGD(S, R) = \frac{\sum_{r \in R} d(r, S)}{|R|} \tag{3}$$

where |R| is the cardinality of R and d (r, S) is the minimum Euclidean distance between r and the points in S. It is important to note that perhaps the elements in R should really be spread evenly, and |R| should be large enough to ensure that the points in R fairly reflect the PF. This ensures that the IGD value of S may accurately assess the solution set's confluence and diversification. S has a lower IGD value, which indicates that it is of higher quality [14].

A set R of indicative points of the PF must be provided in this section to calculate the IGD value of a result set S of a MOEA executing on a MOP.

4.2 Hypervolume indicator

The hyperbolic quantity indicator $I_{hyp}(\mathcal{A})$ calculates the volume of a territory H that is composed of a set of points A and a set of reference points N:

$$I_{\text{hyp}}(\mathcal{A}) = \text{volume}\left(\bigcup_{\forall \boldsymbol{a} \in \mathcal{A}; \forall \boldsymbol{n} \in \mathcal{N}} \text{hypercube}(\text{a},\ \text{n})\right) \tag{4}$$

As a result, higher indicative values correspond to better solutions. The S metric and the Lévesque measure are other names for the hyperdensity indicator. It has a number of appealing attributes that have aided in its adoption and success. It is, in example, the only marker with metric features and the only one that is strictly Pareto monotonic [15]. Because of these characteristics, this indicator has been employed in a variety of applications, including measuring performance and evolutionary programming.

5. Analysis results

Tests and access points for the best algorithm will be presented using a good statistical test called the Wilcoxon Proficient Placement Test Scale.

5.1 Wilcoxon marked

Positional evaluation of the Wilcoxon marked positioning test determines the difference between two illustrations [16] and provides an optional territory trial that is influenced by the sizes and indications of these distinctions. The following theories are addressed by this test:

$$\begin{aligned} H0 &: mean(A) = mean(B) \\ H1 &: mean(A) \neq mean(B) \end{aligned} \tag{5}$$

The solutions to the first and second hypothesis are denoted by the letters A and B, correspondingly. Furthermore, this metric determines if one prediction outperforms the other. Let di denote the gap between the presentation scores of two calculations when it comes to dealing with the *ith* out of n difficulties. Let R^{+} represent the number of sites for instances where the main computation beats the second. Finally, let R- deal with the number of places for the instances where the next estimate outperforms the previous. Several 0's are equitably spread across the entireties. If any of these totals have an odd number, one of them has been discarded:

$$\begin{aligned} \mathrm{R}^{+} &= \sum_{\mathrm{d_i} > 0} \mathrm{rank}(\mathrm{d_i}) + \frac{1}{2} \sum_{\mathrm{d_i} = 0} \mathrm{rank}(\mathrm{d_i}) \\ \mathrm{R}^{-} &= \sum_{\mathrm{d_i} < 0} \mathrm{rank}(\mathrm{d_i}) + \frac{1}{2} \sum_{\mathrm{d_i} = 0} \mathrm{rank}(\mathrm{d_i}) \end{aligned} \tag{6}$$

We utilize MATLAB to find p self-worth in order to contrast the equations at a large degree of alpha = 0.05. Also rand (di) represents the random number between the interval (0, 1).

The invalid hypothesis is rejected when the p-esteem is not exactly the essential part. R+ deals with a high mean estimate that demonstrates predominance over processes of planning using a variety of test setups. This method outperforms all other algorithms in all tests. While $R^{+} = \frac{n*(n+1)}{2}$ surpasses all other techniques in all of adventure.

6. Results and discussion

This section is dedicated to describing and confirming which algorithms are the best in comparison. And the proposed multi-target bat computation (MaBAT/R2) with decay was implemented in Matlab, depending on the problem imposed. The proposed method has been tested with a variety of items, including community size (n), number of iterations, and rate of access reduction β.

The results were applied to fit the proposed methodology for balancing convergence and diversity. On the other hand, we compared MaBAT/R2 with two multi-target PSO accounts to get and know its severity and power to reach the optimal solution. MOPSO [13] and MOEA/D [10] are two different methods. Each calculation is repeated several times in order to achieve the metrics (IGD) and (HV) for each test work. **Table 1** show the following results:

Problem	N	M	D	IBEA	BiGE	KnEA	RVEA	MaBAT/R2
DTLZ1	150	5	9	6.8257e−1 (6.33e−2)	8.9840e−1 (5.53e−2)	6.8355e−1 (1.34e−1)	9.6374e−1 (3.02e−4)	9.7496e−1 (1.40e−4)
	200	10	14	9.0336e−1 (3.58e−2)	2.4403e−1 (1.52e−1)	0.0000e−0 (0.00e−0)	9.2711e−1 (3.91e−2)	9.9749e−1 (2.40e−4)
	250	15	19	9.3565e−1 (2.91e−2)	2.7192e−1 (1.98e−1)	1.8867e−6 (1.03e−5)	9.5880e−1 (2.90e−2)	9.9989e−1 (5.96e−5)
	300	20	24	9.5413e−1 (2.15e−2)	2.2710e−1 (2.53e−1)	3.0867e−3 (1.67e−2)	9.6537e−1 (2.70e−2)	1.0000e+0 (1.63e−6)
DTLZ2	150	5	14	7.9481e−1 (4.08e−4)	7.6948e−1 (5.95e−3)	7.8019e−1 (3.76e−3)	7.9323e−1 (4.92e−4)	7.9836e−1 (1.27e−3)
	200	10	19	9.4386e−1 (2.21e−4)	9.4840e−1 (3.12e−3)	9.5777e−1 (3.02e−3)	9.4295e−1 (3.54e−4)	9.6815e−1 (6.75e−4)
	250	15	24	9.9124e−1 (1.05e−4)	9.8878e−1 (9.11e−4)	6.4009e−1 (4.45e−1) =	9.8956e−1 (7.75e−4)	9.9315e−1 (2.37e−4)
	300	20	29	9.9819e−1 (1.68e−4)	9.9712e−1 (3.54e−4)	4.4353e−1 (4.94e−1)	9.9796e−1 (3.07e−4)	9.9860e−1 (2.91e−4)
DTLZ3	150	5	14	3.7945e−1 (2.18e−3)	4.8529e−1 (1.31e−1)	5.0144e−1 (1.48e−1)	2.4283e−1 (3.24e−1)	7.9323e−1 (1.70e−3)
	200	10	19	6.1617e−1 (1.22e−2)	0.0000e−0 (0.00e−0)	0.0000e−0 (0.00e−0)	7.7229e−1 (2.30e−1)	9.4241e−1 (1.38e−3)
	250	15	24	7.3150e−1 (2.42e−2)	0.0000e−0 (0.00e−0)	0.0000e−0 (0.00e−0)	5.5369e−1 (2.77e−1)	9.9062e−1 (4.53e−4)
	300	20	29	7.8536e−1 (4.53e−2)	0.0000e−0 (0.00e−0)	0.0000e−0 (0.00e−0)	8.1717e−1 (2.40e−1)	9.9792e−1 (3.62e−3)
DTLZ4	150	5	14	7.9124e−1 (2.45e−2)	7.7511e−1 (4.60e−3)	7.8581e−1 (3.86e−3)	7.9315e−1 (5.55e−4) =	7.9184e−1 (1.67e−2)
	200	10	19	9.6910e−1 (2.03e−3) =	9.4497e−1 (2.40e−2)	9.5469e−1 (3.05e−3)	9.4337e−1 (3.21e−4) =	9.3291e−1 (1.80e−2)
	250	15	24	9.9357e−1 (2.21e−4) =	7.8809e−1 (2.55e−2)	9.9298e−1 (4.29e−4)	9.9102e−1 (1.00e−4)	9.9126e−1 (1.11e−4)
	300	20	29	9.9858e−1 (7.05e−5)	8.2491e−1 (3.37e−2)	9.9845e−1 (9.00e−5)	9.9866e−1 (3.00e−5) =	9.9865e−1 (1.00e−4)
DTLZ5	150	5	14	1.1598e−1 (2.85e−3) =	1.1421e−1 (4.17e−3)	8.8917e−2 (1.17e−2)	1.1535e−1 (2.98e−3)	1.0512e−1 (1.10e−3)
	200	10	19	8.9600e−2 (2.00e−3)	9.0956e−2 (1.95e−4)	6.1102e−2 (2.51e−2)	9.1144e−2 (1.13e−3)	9.2377e−2 (1.38e−3)
	250	15	24	8.8414e−2 (4.13e−3)	9.0898e−2 (1.19e−4)	1.8175e−2 (3.03e−2)	9.1038e−2 (5.22e−4) =	9.1199e−2 (5.38e−4)
	300	20	29	8.7890e−2 (5.02e−3)	9.0899e−2 (7.79e−5)	1.0933e−2 (2.51e−2)	9.0972e−2 (3.31e−4) =	9.0949e−2 (2.17e−4)
	+/−/=			0/17/3	0/20/0	0/19/1	0/15/5	

Table 1.
The mean and standard deviation of the IGD value of the proposed algorithms and the four recently comparative algorithms IBEA, BiGE, KnEA, RVEA, and MaBAT/R2 on DTLZ (1–5) for 5, 10, 15, and 20 objective problems, where the best value for each test case is highlighted with a bold background.

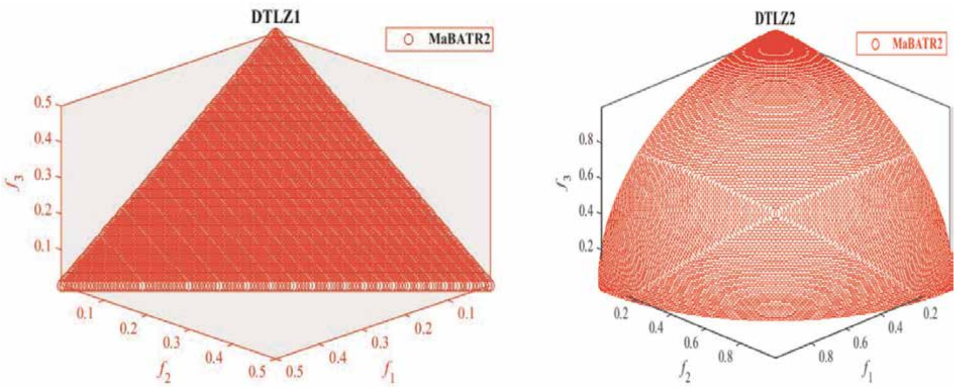


Figure 1.
Number of functions of VS fitness value graph for DTLZ1 and DTLZ2, such that N=No. of population, M = No. of objective function, D = dimension.

7. Convergence graphs

Again for the data sets, an asymptotic graph was constructed to show the speed of convergence of the fitting values with the number of iterations. 100,000 iterations were run for all data. **Figure 1** illustrate this methodologically and analytically effectiveness of the proposed algorithm in obtaining the optimal value as quickly as possible. For this reason, these algorithms were used for comparison: MOEA/D, MOPSO, NSGAII, and SPEA2. All seven algorithms have been applied to 100,000 iterations of hypervolume (HV) and IGD running on them, and their graphs have already been obtained.

8. Conclusions

Many-objective bat algorithms based on deterioration subsystem (MaBAT/R2) are proposed in this paper, in which MOPs are deteriorate into several scalar improvement sub-issues, and each sub-issue is enhanced by just using information from its own few nearby sub-issues in a single run. It is clear from both performance metrics (IGD and HV) that MaBAT/R2 is quite serious and even outflanks the chosen MOBATs. In comparison to the chosen MOBATs, the numbers of Pareto battlefields suggest that MaBAT/R2 can offer quite well Pareto lines.

Additional tests and examinations of the recommended are performed on a case-by-case basis. Later in the project, we will focus on parametric examinations for a broader range of test concerns, including discrete and blended aim of boosting. We aim to examine the various variations of the Pareto frontline it can generate in order to distinguish the methods for improving this computation to meet a range of difficulties. There are a few effective approaches for creating various Pareto fronts, and combining these procedures with others could considerably improve MaBAT/R2.

Acknowledgements

This manuscript is funded by myself.

Author details

Iraq T. Abbas* and Saja Ayad
Department of Mathematics, University of Baghdad, Baghdad, Iraq

*Address all correspondence to: iraq.t@sc.uobaghdad.edu.iq

References

[1] Bosman PA, Thierens D. The balance between proximity and diversity in multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation. 2003;**7**(2):174-188

[2] Yang XS. Bat algorithm for multi-objective optimization. International Journal of Bio-Inspired Computation. 2011;**3**(5):267-274

[3] Laudis LL, Shyam S, Jemila C, Suresh V. MOBA: Multi objective bat algorithm for combinatorial optimization in VLSI. Procedia Computer Science. 2018;**125**:840-846

[4] Remha S, Chettih S, Arif S. A novel multi-objective bat algorithm for optimal placement and sizing of distributed generation in radial distributed systems. Advances in Electrical and Electronic Engineering. 2018;**15**(5):736-746

[5] Talal R. Comparative study between the (ba) algorithm and (pso) algorithm to train (rbf) network at data classification. International Journal of Computer Applications. 2014;**92**(5):16-22

[6] Khan K, Sahai A. A comparison of BA, GA, PSO, BP and LM for training feed forward neural networks in e-learning context. International Journal of Intelligent Systems and Applications. 2012;**4**(7):23

[7] Sheah RH, Abbas IT. Using multi-objective bat algorithm for solving multi-objective non-linear programming problem. Iraqi Journal of Science. 2021: 997-1015

[8] Abbas IT. MOGSABAT: A Metaheuristic Hybrid Algorithm for Solving Multi-Objective Optimization Problems. Springer Nature; 2018

[9] Mirjalili S, Saremi S, Mirjalili SM, Coelho LDS. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. Expert Systems with Applications. 2016;**47**:106-119

[10] Qi Y, Ma X, Liu F, Jiao L, Sun J, Wu J. MOEA/D with adaptive weight adjustment. Evolutionary Computation. 2014;**22**(2):231-264

[11] Li K, Deb K, Zhang Q, Kwong S. An evolutionary many-objective optimization algorithm based on dominance and decomposition. IEEE Transactions on Evolutionary Computation. 2014;**19**(5):694-716

[12] Fleischer M. The measure of Pareto optima applications to multi-objective metaheuristics. In: International Conference on Evolutionary Multi-Criterion Optimization. Berlin, Heidelberg: Springer; 2003. pp. 519-533

[13] Li H, Deb K, Zhang Q, Suganthan PN, Chen L. Comparison between MOEA/D and NSGA-III on a set of novel many and multi-objective benchmark problems with challenging difficulties. Swarm and Evolutionary Computation. 2019;**46**:104-117

[14] Moore J, Chapman R. Application of Particle Swarm to Multi-Objective Optimization. Department of Computer Science and Software Engineering, Auburn University, Tech Science Press; 1999

[15] Peng G, Fang YW, Peng WS, Chai D, Xu Y. Multi-objective particle optimization algorithm based on sharing–learning and dynamic crowding distance. Optik. 2016;**127**(12):5013-5020

[16] Hennequin S, Restrepo LMR. Fuzzy model of a joint maintenance and production control under sustainability constraints. IFAC-Papers Online. 2016;**49**(12):1216-1221

Chapter 7

Intelligent Local Search Optimization Methods to Optimal Morocco Regime

Karim El Moutaouakil, Chellak Saliha, Baïzri Hicham and Cheggour Mouna

Abstract

In this paper, we compare three well-known swarm algorithms on optimal regime based on our mathematical optimization model introduced recently. Different parameters of this latter are estimated based on 176 foods and on who's the nutrients values are calculated for 100 g. The daily nutrients needs are estimated based on the expert's knowledge. Different experimentations are realized for different configurations of the considered swarm algorithms. Compared to Stochastic Fractal Search (SFS) and Particle Swarm Optimization Algorithm (PSO), the Firefly Algorithm (FA) produces the main suitable regimes.

Keywords: optimal regime, favorable nutrient, unfavorable nutrient, quadratic optimization, stochastic fractal search, firefly algorithm, optimization swarm algorithm

1. Introduction

For healthy individuals, balanced diets reduce the likelihood of developing chronic diseases; whereas for individuals with chronic diseases, balanced diets reduce the likelihood of entering dangerous stages, especially for diabetics, cardiovascular disease, obesity and cancer [1–6]. It is a matter of satisfying the body's demands in an optimal manner.

The earliest optimization model, relating to the diet issue, was suggested in [7] with the regime cost as an objective function. Within [8], the target function was minimization of weighted meal compositions, implicating case- and rule-based reasoning; in which any new daily vegan menu consisted of breakfast, lunch, dinner, a snack, and, in additional, a fruit serving. Further suggestions [9] involve minimizing the difference between the real and advised consumption whilst satisfying the nutritional needs. In studies [10], the authors suggest supplemental plans (children under the age of 2 years) and dietary plans (school age group 13–18 years) at the lowest total cost. To further investigate more features, various multi-objective driven schemes were suggested. While generating food meals, the authors of [11] tackled the economical and aesthetical aspects (taste, flavor, color …). When forming the objective functions of their mathematical optimization model, the authors of the article [12] included the price of regime, and other aspects like carbon dioxide emissions, land, and water consumption, etc. V. Mierlo have considered nearly the identical case by substitution of the regime cost and the fossil fuel depletion minimization [13]. At [14], the authors suggest a multi-objective programming framework which delivers a nutritional program plan and minimizes glycemic load and cholesterol consumption, seen as the major causes of childhood overweight.

Recently, we have proposed an original mathematical optimization model for the optimal diet problem. In this paper, we compare three well-known swarm algorithms on optimal regime based on our mathematical optimization model introduced recently [5]. Different parameters of this latter are estimated based on 176 foods who's the nutrients values are calculated for 100 g. The daily nutrients needs are estimated based on the expert's knowledge [6].

The remainder of the material is structured as follows: the second section concerns the mathematical model of the diet problem. The third section is about the three swarm optimization methods: SFS, FA, and PSO. In the fourth section, several experimental results are presented and analyzed. At the end, some conclusions and future propositions are discussed.

2. Optimal regime mathematical model

The quadratic optimization problem which permits the control the total glycemic load of the regime, the lack of positive nutrients, and overdose of negative nutrients in the regime is given by the coming Equations [5, 6, 15]:

$$(D) : \begin{cases} \text{Min}\, g^{\mathsf{T}}x + \theta\ \text{dist}(Ax, b) + \sigma\ \text{dist}(Ex, f) \\ \textit{Subject to}: \\ c_i^{\mathsf{T}}x \geq \rho_i(C^t x) \quad,\ j \in \{car, p\} \\ c_j^{\mathsf{T}}x \leq \tau_j(C^t x),\ \ j \in \{tf, sf\} \\ \mathrm{x} \in [0\ 6]^{176} \end{cases} \tag{1}$$

In the problem (D), $\rho_{car} = 0.55$, $\rho_p = 0.18$, $\tau_{tf} = 0.29$, and$\tau_{sf} = 0.078$ represent the ratios recommended by WHO [16]; g represents the matrix of glycemic load of foods taking into account possible variations; A symbolizes the knowledge of foods in terms of positive nutrients; E gives the amount of negative nutrients in foods;f and b are the daily requirements of positive and negative nutrients, respectively; C is the vector of the foods calories extracted from A; c_{car}, c_p, c_{tf}, and c_{sf} are the calories from carbohydrate, potassium, total fat, and satured fat, respectively. Finally, θ and σ are parameters to control different components of the cost function.

In the Section 4, we will use three optimization swarm algorithms to estimate the optimal diet based on our model for different configurations.

3. Principles and complexity of firefly local search algorithm

This part concerns a brief description of the smart local search optimization methods, called firefly algorithm, we used to solve the diet problem (P).

Firefly algorithm: The Firefly Algorithm (FirA) was originally pioneered by Xin-She Yang [17, 18], on the basis of flashing and behavior models of fireflies. Essentially, FA employs three rules:

a. Fireflies are single-gender and a firefly might be attracting another firefly whatever its gender.

b. Attraction is directly correlated to brightness. If two fireflies are blinking, the darker one will move closer to the lighter one. If there is no firefly with more light, then a random firefly will change its place.

c. The luminosity of a firefly is decided based on the cost function of the problem to be solved.

Because the attractiveness of a firefly is shown to be proportional to the brightness seen by nearby fireflies, given to firefly i and j, the variability of attractiveness δ_{ij}, given the distance d_{ij}, is given by:

$$\delta_{ij} = \delta_0 \exp\left(-\sigma d_{ij}^2\right) \tag{2}$$

Where δ_0 is the basic attracness and σ is a parameter chosen by the user and σ can be chosen based on the formula $\sigma = \sqrt{L}^{-1}$, such that L depends on the large scale of the problem.

Given the current position of the ith x_i^t and jth x_j^t fireflies and the distance between these particles, noted d_{ij}, the position of the ith firefly is updated by:

$$\mathrm{x}_\mathrm{i}^{\mathrm{t}+1} = \mathrm{x}_\mathrm{i}^\mathrm{t} + \delta_{\mathrm{ij}}\left(\mathrm{x}_\mathrm{j}^\mathrm{t} - \mathrm{x}_\mathrm{i}^\mathrm{t}\right) + \alpha_\mathrm{t}\varepsilon_\mathrm{i}^\mathrm{t} \tag{3}$$

The **Figure 1** illustrates the behavior of the ith firefly considering the nearest strong firefly; The random term permits to explore more regions.

α_t is a global random serie of parameters and ε_i^t is personalized local random serie of of parameters linked to the ith firefly. The **Figure 2** gives different steps of the FA algorithm.

Parameters: A good way to control the algorithm randomness is consists on updating α_t based on the formula $\alpha_t = \alpha_0 a^t$ where $a \in [.95.97]$; α_0 represents the initial randomness control factor [18] and can be chosen using the formula $\alpha_0 = .001L$.

Complexity: Considering the two loops of FA, the complexity at the extreme case is $O\left(N^2T\right)$, where N is the number of generated individuals and T is the number of

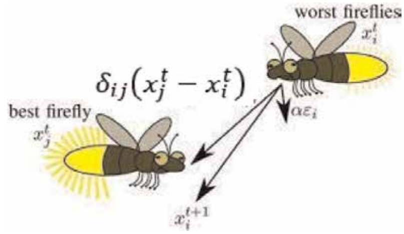


Figure 1.
Ullistration of the behavior of the ith firefly considering the nearst strong firefly.

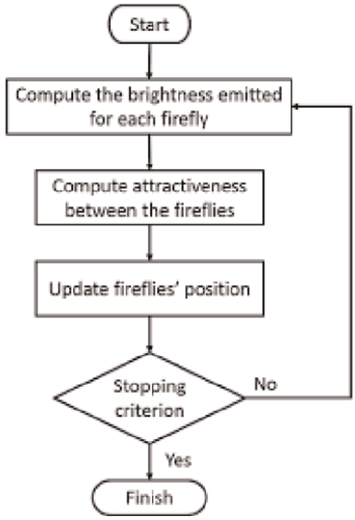


Figure 2.
Diagram of the FA algorithm.

iterations. To reduce the complexity of FA, we can rank the attractiveness or brightness using sorting algorithms and the complexity becomes $O(TNlog(N))$.

Variants: In the case of combinatorial optimization Problems, variants of FA were developed with improved efficiency [19–21].

4. Stochastic fractal search algorithm

SFS is inspired by the background process of development. This algorithm is a computational search method that utilizes a mathematical principle known as a fractal [22]. Fractal search uses 3 rules to come up with a solution: (a) every particle has an electrical potential energy, (b) every particle spread and induces the generation of more random particles, and the starting particle's energy is shared among the newly formed particles, and (c) just a small amount of the better particles stay in the next round, and the remaining particles are skipped. The **Figure 3** illustrates the diffusion

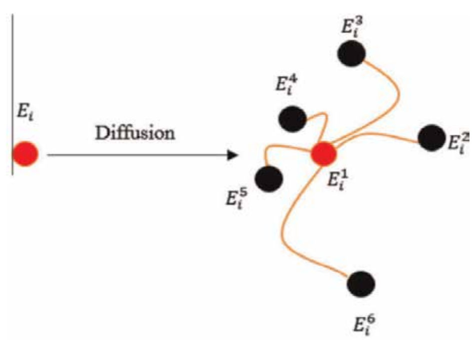


Figure 3.
Particle diffusion.

of the particle E_i . This strategy works well in identifying the solution; however, the method has its drawbacks.

The major problem is the high number of parameters required to be properly managed, and the additional issue is that the interchange of knowledge is not taking place between the individual. To overcome the above challenges, Salimi, H. introduced another version of fractal search called stochastic fractal search [22].

In the SFS algorithm, two main operations take place: the diffusion operation and the updating operation. In the first operation, each particle scatters around its current position to satisfy the intensification (exploitation) property. In the latter operation, the algorithm mimics the way an individual updates his location depending on the position of the remaining individual in this cluster.

To generate new individual from the scattering operation, Lévy and Gaussian flight are investigated as two statistical methods. Generally, a sequence of Gaussian treads participating in the scattering operation were listed in the next equations:

$$\mathrm{GW}_1 = \mathrm{N}(\mu_{\mathrm{BP}}, \sigma) + \varepsilon \mathrm{BP} - \varepsilon' \mathrm{P_i} \text{ and } \mathrm{GW}_2 = \mathrm{N}(\mu_{\mathrm{P}}, \sigma) \tag{4}$$

Here $\varepsilon, \varepsilon' \sim U([0\ 1])$, BP denotes the global best position, P_i is the position of the current particle, $\mu_{BP} = BP$, $\mu_P = P_i$, and σ is given by $\sigma = \left|\frac{\log(g)}{g}(P_i - BP)\right|$; g represents the number of iterations and $\frac{\log(g)}{g}$ permits to reduce the size of the normal step.

To ensure a good exploration of the research domain, two statistical strategies are considered:

(a) A uniform probability weight is attributed to each individual i in the group:

$$Pa_i = \frac{\textit{the rank of the point i in the group}}{\textit{the number of the points in the group}} = \frac{rank(P_i)}{N}. \tag{5}$$

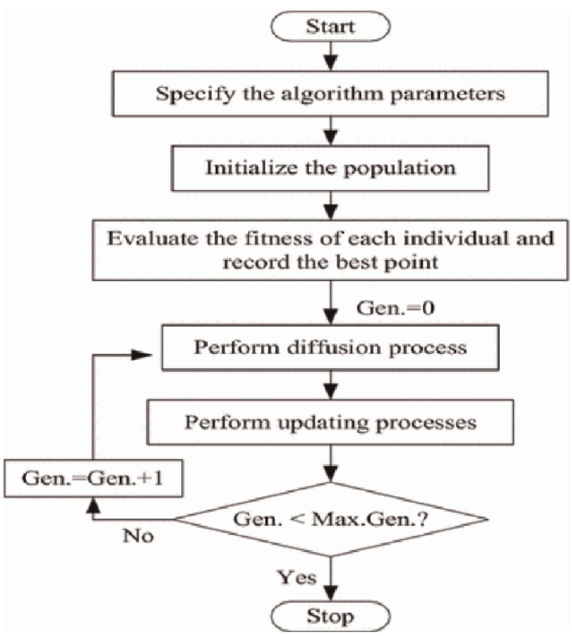


Figure 4.
Diagram of SFS algorithm.

In this sense, Pa_i is less than a given threshold, the position of the ith point, from the group G, is updated using the equation:

$$P'_i = P_{rand1\_G} - \varepsilon(P_{rand2\_G} - P_i) \text{ such that } \varepsilon \sim U([0\ 1]) \tag{6}$$

As in the first process, if the $Pa_i \leq \varepsilon$ holds, the current particle is changed:
If $\varepsilon' \leq .5$, then $P''_i = P'_i - \hat{\varepsilon}\left(P'_{rand1\_G} - PB\right)$, else $P''_i = P'_i - \hat{\varepsilon}\left(P'_{rand1\_G} - P'_{rand2\_G}\right)$,
Where $\hat{\varepsilon} \sim U([0\ 1]$
The **Figure 4** illustrates different steps of SFS algorithm; for more details, the reader can see the paper of Salimi [22].

5. Particle swarm algorithm

Particle swarm optimization, first introduced by Kennedy and Eberhart [23], is a synthetic meta-heuristic approach to global computer optimization, belonging to the swarm intelligence concept-based algorithm family of approaches.

5.1 Basic PSO algorithm

Each potential solution is known as a "particle" within PSO and the location of the ith particle may be determined by $\boldsymbol{p_i} = \left(\boldsymbol{p_{ij}}\right)_{\boldsymbol{j=1,\dots,n}}$ where n is the dimension of the search space. From now on, we suppose that we have a swarm P of N particle $\boldsymbol{p_1}, \dots, \boldsymbol{p_N}$.

During the search process, the particles update their positions using the motion equation:

$$\mathrm{p}_\mathrm{i}^{\mathrm{t+1}} = \mathrm{p}_\mathrm{i}^\mathrm{t} + \mathrm{v}_\mathrm{i}^{\mathrm{t+1}} \tag{7}$$

The ith particle velocity is given by:

$$\mathrm{v}_\mathrm{i}^{\mathrm{t+1}} = \mathrm{v}_\mathrm{i}^\mathrm{t} + \mathbf{c}_1\left(\mathrm{bp}_\mathrm{i} - \mathrm{p}_\mathrm{i}^\mathrm{t}\right)\mathbf{r}_1 + \mathbf{c}_2\left(\mathrm{bg} - \mathrm{p}_\mathrm{i}^\mathrm{t}\right)\mathbf{r}_2 \tag{8}$$

Such that $\boldsymbol{bp_i}$ is the best position of the particle i, g is the global best position of the swarm members, $\boldsymbol{c_k}, k = 1, 2$, is the acceleration parameters usually thoken from the interval [0 4] named also "cognitive coefficient", and $\boldsymbol{r_k} = \mathrm{diag}(\mathrm{uniform}([0\ 1])), k = 1, 2$. The **Figure 4** illustrates the PSO formula used to update the particles positions (**Figure 5**).

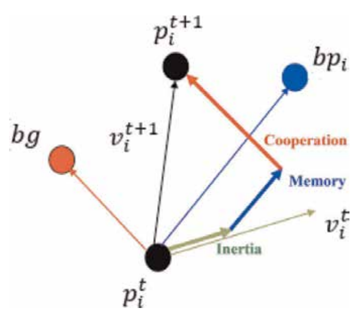


Figure 5.
PSO learning equation illustration.

The basic PSO pseudo-code can be the following:

1. Initialization. For each of the N particles:

   a. Initialize the position p_i^0;

   b. Initialize the particle's best position to this initial position $bp_i^0 = p_i^0$;

   c. Calculate the fitness of each particle and $\mathrm{bg} = p_j^0$ with $f\left(p_j^0\right) \geq f\left(p_i^0\right)$.

2. Repeat the coming steps until convergence:

   a. Update the velocity using:

$$\mathrm{v}_\mathrm{i}^{\mathrm{t}+1} = \mathrm{v}_\mathrm{i}^\mathrm{t} + \mathbf{c}_1\left(\mathrm{bp}_\mathrm{i} - \mathrm{p}_\mathrm{i}^\mathrm{t}\right)\mathbf{r}_1 + \mathbf{c}_2\left(\mathrm{bg} - \mathrm{p}_\mathrm{i}^\mathrm{t}\right)\mathbf{r}_2 \tag{9}$$

   b. Update the particle position using:

$$\mathrm{p}_\mathrm{i}^{\mathrm{t}+1} = \mathrm{p}_\mathrm{i}^\mathrm{t} + \mathrm{v}_\mathrm{i}^{\mathrm{t}+1} \tag{10}$$

   c. Evaluate the ith particle fitness $f\left(p_i^{t+1}\right)$;

   d. If $f\left(p_i^{t+1}\right) \geq f\left(p_i^{t+1}\right),; bp_i = p_i^{t+1}$

   e. If $f\left(p_i^{t+1}\right) \geq bg),; bg = p_i^{t+1}$

3. At the convergence the best solution is bg.

5.2 PSO meta parameters

Initialization: PSO involves an initial estimate of the positions and velocities. For the initial positions, a general consensus is to cover the solution space on a uniform basis: $p_{ij}^0 \sim U\left(\left[LB_j, UB_j\right]\right)$. For initial velocities, it is suggested to use a uniform distribution to ensure a uniform coverage of the search space. But this could augment the probability of particles being infeasible solutions. To defeat this inconvenience, the velocities may be set to zero or to very tiny arbitrary numbers.

Acceleration constants: The parameters c_1 and c_2 have a very large impact on the particle's paths and on the algorithm convergence. In this sense, the larger these constants are, the more the oscillation of the particle around the optimum increases, whereas very small values give rise to sinusoidal patterns. In general, it is recommended to set these parameters to 2 [24].

Swarm size: A large swarm size improves the variety of the swarm and its exploration ability, but in another way, it may also increase the risk of an early convergence and the calculation costs. Nevertheless, in most situations, it has actually been found that once the swarm size is higher than 50 particles, PSO becomes insensitive to the swarm size [24].

6. Experimentation and analysis

We utilize FA, PSO, and SFS algorithms to establish optimal regimes based on the proposed mathematical model in [5] where $\theta = 0.67$ and $\sigma = 1.34$. The WHO recommendations concerning the nutrients daily needs were token into considerations [6, 25, 26]. We work on 176 aliments considered as the most consumed in Morocco. The linear part of our model is estimated using the means glycemic load of the considered foods. From now on, we adopt the symbols: TGL for Total Glycemic Load, FTG for Favorable Totale Gap, and UFTG for UFavorable Totale Gap.

a. We used the SFS algorithm to solve problem (D). We tested this algorithm for different values of the parameters: walk probability, maximum diffusion, and the number of iterations. Th **Table 1** gives TG, FTG, and UFTG of diets produced by SFS for max diffusion equals to 5, start points equals to 50, number of iterations of 200, and different values of walk probability from the interval [0.3 0.9] adopting 0.1 as step.

The best diet is the one produced by SFS for walk probability value equals to 0.7 with glycemic load in the interval [82.2152 92.5292] and nutrients requirements gaps

SFS walk probability	Diet total glycemic load			FTG	UFTG
	min	mean	max		
0.3	95.4564	116.5515	120.6926	210.7442	18.9232
0.4	87.0935	94.1033	97.3494	190.9682	30.2345
0.5	89.6853	97.8046	98.4658	216.7432	37.7244
0.6	101.0979	113.8293	110.0329	170.6037	22.8605
0.7	**82.2152**	**87.4453**	**92.5292**	**143.3103**	**30.9554**
0.8	86.3963	94.3195	94.5612	151.0829	30.5128
0.9	86.3344	94.2951	99.5339	164.5838	46.2813

Table 1.
TG, FTG, and UFTG of diets produced by SFS for max diffusion = 5, start points = 50, number of iterations of 200, and different values of walk probability.

SFS diffusion	Diet total glycemic load			FTG	UFTG
	min	mean	max		
5/45	76.9257	82.9567	83.2560	133.2240	40.1505
5/50	82.2152	87.4453	92.5292	143.3103	30.9554
6/45	73.0082	84.9846	87.1250	196.1147	48.0560
7/45	94.8373	98.9290	101.1763	77.5158	6.1873
9/45	84.0789	99.0630	100.9079	106.8640	18.2439
10/45	**68.8041**	**74.0633**	**76.3373**	**52.0240**	**47.8260**

Table 2.
TG, FTG, and UFTG of diets produced by SFS for start points = 50 (45), number of iterations of 200, walk probability of 0.7, and different values of diffusion.

143.3103 mg (for positive nutrients) and 30.9554 mg (for negative nutrients). These diets still bad considering the considered three criterions. To investigate possible improvements, we set the walk probability to 0.7 and, start points to 45, and number of iterations to 200, and we variate the value of diffusion.

The **Table 2** give TGL, FTG, and UFTG of diets produced by SFS for start points equals to 50(45), number of iterations of 200, walk probability of 0.7, and different values of diffusion from [5 10] by adopting 1 as step. The obtained diets become to be acceptable and the best diet is the one who's TGL is in [**68.8041 76.3373**], FTG = **52.0240**, and UFTG = **47.8260.**

To investigate more improvements, we set max diffusion to 10, walk probability to 0.7, start points to 45, and we vary different number of iterations; see **Table 3**.

Indeed, we detect a very good diet (produced by SFS) for 600 number of iterations with TG is in [53.8780 66.0715], FTG = 50.1917, and UFTG = 28.5891. The **Figure 6** illustrates the behavior of (D) objective function when solving the diet problem using SFS for max diffusion equals to10, walk probability equals to 0.7, start points equals to 45, and the number of iterations equals to 600; it is clear that the algorithm has not yet

SFS iterations number	The diet total glycemic load			FTG	UFTG
	min	mean	max		
300	68.8041	74.0633	76.3373	52.0240	47.8260
400	91.5940	95.7307	98.1429	86.6777	10.5893
500	82.1371	88.2860	93.6195	119.4418	12.2939
600	**53.8780**	**60.5048**	**66.0715**	**50.1917**	**28.5891**
700	79.8448	82.8554	85.6542	68.4567	16.4858
800	84.6661	95.5110	105.3748	23.6773	21.2563

Table 3.
TG, FTG, and UFTG of diets produced by SFS for max diffusion 10, walk probability 0.7, start points 45, and different number of iterations.

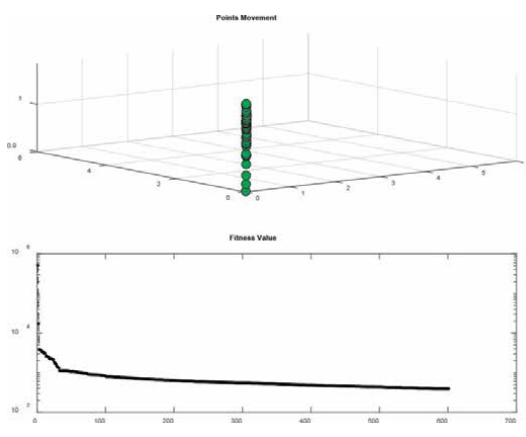


Figure 6.
Evolution of the model (D) fitness with iterations by SFS for walk probability of 0.7, maximum diffusion of 10, and number of iteration equals to 600.

converged and an additional number of iterations will allow more improvement, but we compare the algorithms for a very small number of iterations to get a good diet in real time.

b. We used the FA algorithm to solve problem (D). We tested this algorithm for different values of the parameter's population, attraction coefficient base value, iterations, and of Mutation coefficient damping ratio.

The **Table 4** give TG, FTG, and UFTG of diets produced by FA for: population 40, attraction coefficient base value of 2.25, iterations of 300, variation of mutation coefficient damping ratio from in [0.1 0.9] with 0.1 as step.

All the produced diets are acceptable and the best diet is the one produced for Mutation Coefficient Damping Ratio equals to 0.4. To investigate more improvements

FA Mutation Coefficient	Diet total glycemic load			FTG	UFTG
	min	mean	max		
0.1	52.6591	53.0527	53.4451	32.7860	3.1337
0.2	52.9493	54.3853	55.8213	10.0024	12.5275
0.3	77.3769	78.5940	79.7333	10.0022	11.3670
0.4	**68.7673**	**71.0372**	**73.1670**	**14.8004**	**4.7598**
0.5	69.6771	70.7284	71.7793	19.9331	7.4906
0.6	59.7460	61.6316	63.3772	5.1746	19.7053
0.7	64.4724	65.4970	66.3804	16.5202	20.7693
0.8	59.0634	60.0857	61.0316	12.5387	1.0147
0.9	63.6272	64.4156	65.0629	11.0490	18.8717

Table 4.
Diet produced by FA for population equals to 40, attraction coefficient base value of 2.25, iterations equals to 300, and variation of mutation coefficient damping ratio.

FA population size	Diet total glycemic load			FTG	UFTG
	min	mean	max		
20	64.2545	65.9005	67.5535	13.5829	32.0750
25	42.2206	43.8853	45.4469	56.8449	7.4848
30	54.6090	56.0057	57.3672	72.9725	5.3529
35	79.2812	81.8633	84.3053	3.0136	15.7434
40	68.7673	71.0372	73.1670	14.8004	4.7598
45	76.3777	78.0148	79.6519	26.6040	3.2404
50	**53.5439**	**54.9658**	**56.3875**	**10.6000**	**7.8365**

Table 5.
Diets produced by FA for attraction coefficient base value equals to 2.25, iterations equals 300, mutation coefficient damping ratio = 0.4, and variation of population.

of this diets, we variate the number of iterations will setting the mutation coefficient damping ratio to 0.4; see **Table 5**.

In fact, the quality of diets were improveded and the best one is obtained for attraction coefficient base value equals to 2.25, iterations equals to 300, mutation coefficient damping ratio equals to 0.4, size population = 50 with TGL is in [**53.5439 56.3875**], FTG = **10.6000 mg**, and UFTG = **7.8365 mg.**

The **Figure 7** illustrates the behavior of (D) objective function when solving the diet problem using FA for coefficient base value equals to 2.25, iterations equals to 200, mutation coefficient damping ratio equals to 0.4, and size of population equals to 50. We remark that FA algorithm reaches early a very good local solution.

c. We used the PSO algorithm to solve problem (D). We tested this algorithm for different values of iterations, self-adjustment weight, social-adjustment weight, and population size.

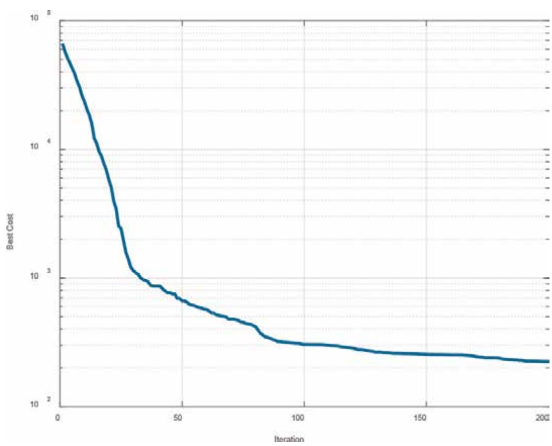


Figure 7.
Behavior of (D) objective function when solving by FA for: Firefly attraction coefficient base value = 2.25, iterations = 200, mutation coefficient damping ratio = 0.4, variation of population = 50.

PSO population size	Diet total glycemic load			FTG (mg)	UFTG (mg)
	min	mean	max		
20	59.6241	63.5302	68.5703	170.9513	22.2901
30	75.4830	85.8035	88.7775	170.4762	77.3999
40	68.5199	78.5372	80.0771	102.6863	36.4461
50	**70.8154**	**75.5430**	**80.1564**	**61.6584**	**19.1466**
60	83.5883	91.8249	98.7967	469.1408	184.4479
70	72.0223	81.4192	81.8198	170.7936	29.8207
80	69.5418	74.5632	80.3492	133.7875	37.3952

Table 6.
Diets produced by PSO for number of iterations = 200, self-adjustment weight = social-adjustment weight = 2, and variation of the population size.

PSO Adjustment Weight	Diet total glycemic load			FTG (mg)	UFTG (mg)
	min	mean	max		
1	89.4023	100.8713	111.2694	1.3158e+03	122.9172
1.1	87.1844	95.9035	98.1930	1.0457e+03	79.9678
1.2	74.1234	83.6180	84.1389	230.2951	482.5612
1.3	81.7219	91.7305	94.7066	117.2526	55.7460
1.4	81.9234	94.7501	101.9684	236.6845	45.4220
1.5	70.6824	82.5010	85.0437	551.0106	52.5807
1.6	77.2499	87.0125	89.6039	116.2240	61.8163
1.7	73.5639	79.0021	83.3030	66.6849	25.3209
1.8	58.8132	60.9824	62.9015	110.0112	57.2320
1.9	71.0009	73.7154	79.7176	146.4381	127.1389
2	**70.8154**	**75.5430**	**80.1564**	**61.6584**	**19.1466**

Table 7.
Diets produced by PSO for number of iterations = 200, variation of self-adjustment weight = SocialAdjustmentWeight, and population size =50.

The **Table 6** give TG, FTG, and UFTG of diets produced by FA for number of iterations equals to 200, self-adjustment weight = social-adjustment weight = 2, and population size variation between 20 and 80 particles.

The best diet is the one produced by PSO for population size of 50 with TG in [70.8154 80.1564], FTG = 61.6584 mg, and UFTG = 19.1466 mg. To investigate more improvements of this diets, we vary the Adjustment Weight coefficients in [1 2] will setting the population size to 50 (**Table 7**).

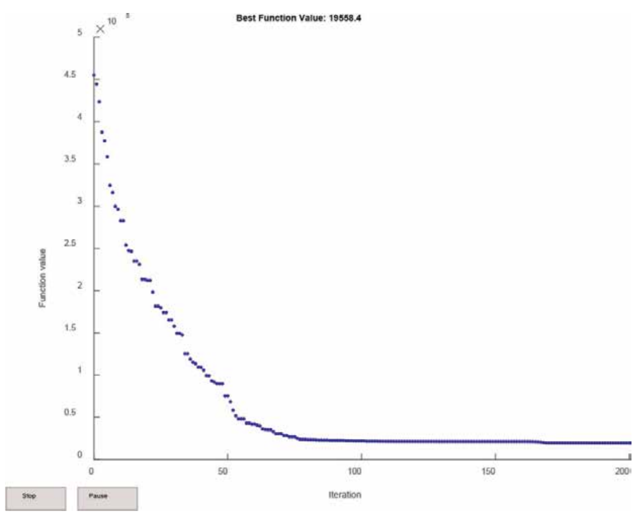


Figure 8.
The behavior of (D) objective function when solving the diet problem using PSO for number of iterations = 200, self-adjustment weight = social-adjustment weight = 2, and population size = 50.

Method	Parameters values	Diet total glycemic load			FTG (mg)	UFTG (mg)
		min	mean	max		
SFS	• Walk probability = 0.7 • Diffusion = 45 • Maximum diffusion = 10 • Number of iteration = 600	53.8780	60.5048	66.0715	50.1917	28.5891
FA	• Attraction coefficient base value = 2.25, • Iterations = 300, • Mutation coefficient damping ratio = 0.4, • Variation of population = 40	53.5439	54.9658	56.3875	10.6000	7.8365
PSO	• Iterations = 200, • Adjustement weight = 2, • Population size = 50	70.8154	75.5430	80.1564	61.6584	19.1466

Table 8.
Comparison between the diets produced by PSO, FA, and SFS.

Indeed, the quality of diets were improveded and the best one is obtained for PSO with iterations = 200, variation of self adjustment weight = social adjustment weight = 2, and population size =50; the Diet total glycemic load is in [70.8154 80.1564] and FTG = 61.6584 mg, and UFTG = 19.1466 mg, which meets the recommandations given in [24] . It should be noted that the first height diets are unacceptable.

The **Figure 8** illustrates the behavior of (D) objective function when solving the diet problem using PSO for self-adjustment weight = social-adjustment weight = 2, and population size =50. We remark that PSO was attracted very early to a very bad diet.

d. We compared the best diets produced by SFS, FA, and PSO based on the considered three criteria: TGL, FTG, and UFTG; see **Table 8**.

We remark that the best diet is the one produced by firefly algorithm for the configuration shown by the column 2 of the **Table 8** for a small number of iterations.

We can repeat all this study will consider additional quality measures such as the satiety rate and the applicability of the considered diets.

7. Conclusion

In this work, we used well-known swarm algorithms to solve the optimal diet problem based on the optimization mathematical model proposed recently in [5]. The inputs of our model were estimated based on 176 Morocco foods. Based on different paper search and the WHO's recommendations, we have estimated the daily nutrients requirements [6]. Different experimentations were realized for different configurations of the considered algorithms. Concerning SFS algorithm, we solved the problem (D) for different values of walk probability (0.7*), maximum diffusion (10*), and number of iteration (600*). Concerning FA algorithm, we solved the problem (D) for different values of attraction coefficient base value (2.25*), iterations (300*), mutation coefficient damping ratio (0.4*), and variation of population (40*). Concerning PSO, we solved the problem (D) for different values of Iterations (200*), adjustment weight (2*), and population size (50*). The best diets were produced by Firefly algorithm.

We can replicate that this entire investigation will consider further metrics of quality like satiety rate and feasibility of the examined diets.

In the future, we will propose a hybrid algorithm based on the SFS, FA, and PSO; this algorithm will be used to solve the diet problem and other well-known problems.

Acknowledgements

This work was supported by Ministry of National Education, Professional Training, Higher Education and Scientific Research and the Digital Development Agency (DDA) and CNRST of Morocco (Nos. Alkhawarizmi/2020/23).

Author details

Karim El Moutaouakil[1*], Chellak Saliha[2], Baïzri Hicham[3] and Cheggour Mouna[4]

1 Faculty Polydisciplinary, Engineering Science Laboratory, USMBA, Taza, Morocco

2 MorphoSciences Research Laboratory of the Faculty of Medicine and Pharmacy of Marrakech, Université Cadi Ayyad, Marrakech, Morocco

3 Faculty of Medicine and Pharmacy Université Cadi Ayyad, Hicham Baïzri The Biosciences and Health Laborator, Marrakech, Morocco

4 FMPM, Cadi Ayyad University, Marrakech, Morocco

*Address all correspondence to: karim.elmoutaouakil@usmba.ac.ma

References

[1] Fukuoka Y, Komatsu J, Suarez L, Vittinghoff E, Haskell W, Noorishad T, et al. The mPED randomized controlled clinical trial: Applying mobile persuasive technologies to increase physical activity in sedentary women protocol. BMC Public Health. 2011;**11**(1):1-8

[2] Chen XC, Wei TR, Guo JH, Zhou WX, Song A, Chen WN, et al. August. Multiobjective evolutionary algorithm with double-level archives for nutritional dietary decision problem. In: 2019 9th International Conference on Information Science and Technology (ICIST). 2019. pp. 419-426

[3] Mortada S, Abou Najm M, Yassine A, El Fadel M, Alamiddine I. Towards sustainable water-food nexus: An optimization approach. Journal of Cleaner Production. 2018;**178**: 408-418

[4] Seligman HK, Laraia BA, Kushel MB. Food insecurity is associated with chronic disease among low-income NHANES participants. The Journal of Nutrition. 2010;**140**(2):304-310

[5] El Moutaouakil K, Cheggour M, Chellak S, et al. Metaheuristics optimization algorithm to an optimal Moroccan diet. In: 2021 7th Annual International Conference on Network and Information Systems for Computers (ICNISC). IEEE; 2021. pp. 364-368

[6] El Moutaouakil K, Ahourag A, Chellak S, Baïzri H, Cheggour M. Fuzzy deep daily nutrients requirements representation. Revue d'Intelligence Artificielle. 2022;**36**(2):263-269. DOI: 10.18280/ria.360210

[7] Stigler GJ. The cost of subsistence. Journal of Farm Economics. 1945;**27**(2): 303-314

[8] Orešković P, Kljusurić JG, Šatalić Z. Computer-generated vegan menus: The importance of food composition database choice. Journal of Food Composition and Analysis. 2015;**37**: 112-118

[9] Masset G, Monsivais P, Maillot M, Darmon N, Drewnowski A. Diet optimization methods can help translate dietary guidelines into a cancer prevention food plan. The Journal of Nutrition. 2009;**139**(8):1541-1548

[10] Briend A, Darmon N, Ferguson E, Erhardt JG. Linear programming: A mathematical tool for analyzing and optimizing children's diets during the complementary feeding period. Journal of Pediatric Gastroenterology and Nutrition. 2003;**36**(1): 12-22

[11] Seljak BK. June. Dietary menu planning using an evolutionary method. In: 2006 International Conference on Intelligent Engineering Systems. 2006. pp. 108-113

[12] Donati M, Menozzi D, Zighetti C, Rosi A, Zinetti A, Scazzina F. Towards a sustainable diet combining economic, environmental and nutritional objectives. Appetite. 2016;**106**:48-57

[13] Van Mierlo K, Rohmer S, Gerdessen JC. A model for composing meat replacers: Reducing the environmental impact of our food consumption pattern while retaining its nutritional value. Journal of Cleaner Production. 2017;**165**: 930-950

[14] Bello P, Gallardo P, Pradenas L, Ferland JA, Parada V. Best compromise nutritional menus for childhood obesity. PLoS One. 2020;**15**(1):0216516

[15] Bas E. A robust optimization approach to diet problem with overall glycemic load as objective function. Applied Mathematical Modelling. 2014; **38**(19–20):4926-4940

[16] You A. Dietary Guidelines for Americans. USA: US Department of Health and Human Services and US Department of Agriculture; 2015

[17] Yang XS. Nature-Inspired Metaheuristic Algorithms. Luniver Press; 2010

[18] Yang XS. Firefly algorithms for multimodal optimisation. In: Watanabe O, Zeugmann T, editors. Proc. 5th Symposium on Stochastic Algorithms, Foundations and Applications. Vol. 5792. Lecture Notes in Computer Science; 2009. pp. 169-178

[19] dos Santos Coelho L, de Andrade Bernert DL, Mariani VC. A chaotic firefly algorithm applied to reliability-redundancy optimisation. In: 2011 IEEE Congress on Evolutionary Computation (CEC'11). 2011. pp. 517-521

[20] Fister J, Fister I, Brest J, Yang XS. Memetic firefly algorithm for combinatorial optimisation. In: Filipič B, Silc J, editors. Bioinspired Optimisation Methods and their Applications (BIOMA2012). Bohinj, Slovenia; 2012. pp. 75-86

[21] Hassanzadeh T, Vojodi H, Moghadam AME. An image segmentation approach based on maximum variance intra-cluster method and firefly algorithm. In: Proc. of 7th Int. Conf. On Natural Computation (ICNC2011). 2011. pp. 1817-1821

[22] Salimi H. Stochastic fractal search: A powerful metaheuristic algorithm. Knowledge-Based Systems. 2015;**75**: 1-18

[23] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks. Vol. 4. 1995. pp. 1942-1948

[24] Eberhart RC, Shi Y, Kennedy J. Swarm Intelligence. Burlington, MA: Morgan Kaufmann; 2001

[25] Humayun MA, Elango R, Ball RO, Pencharz PB. Reevaluation of the protein requirement in young men with the indicator amino acid oxidation technique. The American Journal of Clinical Nutrition. 2007;**86**(4):995-1002

[26] Verma H, Garg R. Effect of magnesium supplementation on type 2 diabetes associated cardiovascular risk factors: A systematic review and meta-analysis. Journal of Human Nutrition and Dietetics. 2017;**30**(5):621-633

Chapter 8

Design Optimization of 18-Poled High-Speed Permanent Magnet Synchronous Generator

Aslan Deniz Karaoglan, Deniz Perin and Kemal Yilmaz

Abstract

The aim of this research is to optimize the design of an 18-poled 8000 rpm 7 kVA high-speed permanent magnet synchronous generator. The goal is to find the best factor levels for the design parameters, namely magnet thickness (MH), offset, and embrace (EMB) to optimize the responses namely efficiency (%), rated torque (N.m), air-gap flux density (Tesla), armature current density (A/mm^2), armature thermal load (A^2/mm^3). The aim is to keep the air-gap flux density at 1 tesla while maximizing efficiency and minimizing the rest of the responses. Optimization was carried out with one sample algorithm selected from each of the commonly used optimization algorithm classifications. For this purpose, different class of well-known optimization techniques such as response surface methodology (gradient-based methods), genetic algorithm (evolutionary-based algorithms), particle swarm optimization algorithm (swarm-based optimization algorithms), and modified social group optimization algorithm (human-based optimization algorithms) are selected. In the Ansys Maxwell environment, numerical simulations are carried out. Mathematical modeling and optimizations are performed by using Minitab and Matlab, respectively. Confirmations are also performed. Results of the comparisons show that modified social group optimization and particle swarm optimization algorithms a bit outperform the response surface methodology and genetic algorithm, for this design problem.

Keywords: high-speed alternator, permanent magnet synchronous generator, electric machine design, design optimization, response surface methodology, modified social group optimization algorithm, particle swarm optimization algorithm, genetic algorithm

1. Introduction

Many researchers have studied magnetic device design optimization and permanent magnet synchronous generator (PMSG) design optimization, which are investigated in many research studies over the last few decades. Efficiency, magnetic flux density distribution, total harmonic distortion (THD), and other performance criteria are commonly used in these studies and are attempted to be improved [1–11]. The most common problems are heat buildup in the rotor, balancing issues, and bearing issues. Magnetic flux density distribution is a key success criterion that must be maintained within a specific range in order to provide high efficiency and low heating for the electric machine. Many different methods are used for design optimization. It is impossible to perform optimization using real experimental results because there are so many design combinations. In most cases, simulation results are used instead. However, there are limited numbers of studies about high-speed alternator design optimization.

Sadeghierad et al. [12] studied on performance comparisons of alternative designs of high-speed alternators (HSA) for microturbines and considered the design difficulties. Sadeghierad et al. [13] studied on optimizing the design of a high-speed axial flux generator (HSAFG) by the aid of particle swarm optimization (PSO) and genetic algorithm (GA) to maximize the efficiency. They discussed the effect of the lambda, which is the ratio of inner diameter to outer diameter. Ismagilov et al. [14] tested a new topology of the stator magnetic core made of amorphous alloy for a 5 kW 60,000 rpm high-speed permanent magnet electric machine with a tooth-coil winding with six slots and two and four poles. Guo et al. [15] presented a method for determining the back electromotive force (EMF) utilizing air gap static flux density distribution and calculating the coil average inductance at the midline of the quadrature-direct axis. They used gradient descent-based optimization to minimize the volume of high-speed generator for micro turbojet engine. The summary for the state of the art is given in **Table 1**.

As can be seen from the literature review, the studies about design optimization of high-speed generator by using optimization methods are very limited. Also the results those presented to show the performance comparisons of the meta-heuristic optimization methods are very poor.

The motivation of this study is to perform design optimization of 18-Poled 8000 rpm 7 kVA high-speed PMSG. This problem is important because of the high rotor speed and high frequency of the stator flux variation; the design of a high-speed machine differs significantly from the design of a conventional machine with low speed and low frequency. The first motivation of this study is to contribute to the knowledge that has emerged based on the limited number of studies on this subject in the literature, with a new study on topology optimization of high-speed PMSGs.

The second motivation is to show the performance of the different class of optimization techniques on the design problem of high-speed alternators to the related researchers. Deterministic or stochastic algorithms can be used for optimization. Due to their high processing demands, deterministic approaches are ineffective for handling multimodal and nonlinear complex issues. The nature is a major source of inspiration for meta-heuristic algorithms, which are stochastic approaches utilized for

Author(s)	Year	Subject	Optimization method
Sadeghierad et al.	2006	HSA for microturbines	N/A
Sadeghierad et al.	2010	HSAFG	PSO, GA
Ismagilov et al.	2018	5 kW 60,000 rpm high-speed permanent magnet electric machine	N/A
Guo et al.	2019	High speed generator for micro turbojet engine	Gradient descent method

Table 1.
Summary of the literature review.

optimization. There are four different types of meta-heuristics: (*i*) evolutionary, (*ii*) swarm, (*iii*) physical and chemical, and (*iv*) human. Various well-known optimization methods, such as response surface methodology (RSM) (gradient-based methods), GA (evolutionary-based algorithms), PSO (swarm-based optimization algorithms), and modified social group optimization algorithm (MSGO) (human-based optimization algorithms), are used for this purpose.

The selected design parameters (magnet thickness (MH), offset, embrace (EMB)) and the responses (efficiency (%), rated torque (N.m), air-gap flux density (Tesla), armature current density (A/mm^2), armature thermal load (A^2/mm^3)) are not previously used together for high-speed alternator design optimization problem. So this is the novelty aspect of this research.

This research was carried out in a real industrial plant, and by focusing on a small number of parameters, we hoped to have a smaller impact on the layout and operation of a serial production line (Such as redesigning assembly parts that may have an impact on standard production, cooling design, body design, and so on.). As a result, the parameters (magnet thickness (MH), offset, embrace (EMB)) that have the least impact on the outer dimensions of the alternator are chosen as the design parameters (factors). The following section goes over the materials and methods.

2. Materials and methods

2.1 Regression modeling and response surface methodology (RSM)

The design optimization problem that is handled in this study is solved in three steps: *i*) design the experiments and perform the experimental runs, *ii*) perform regression modeling to determine the mathematical relations between the responses and the factors, *iii*) perform optimization to determine the optimum factor levels. The goal of this paper is to calculate the optimum levels of magnet thickness (X_1), offset (X_2), and embrace (X_3) to maximize the efficiency and to minimize the rated torque, armature current density, and armature thermal load, while keeping the air-gap flux density at 1.0 Tesla. Linear, quadratic, and interaction terms can all be found in regression models. These three terms occur simultaneously in a full quadratic model. Eq. (1) provides the full quadratic model's general representation [16–18].

$$Y_i = \beta_0 + \sum_{k=1}^{m} \beta_k X_{ki} + \sum_{k=1}^{m} \beta_{kk} X_{ki}^2 + \sum_{k<l}^{m} \beta_{kl} X_{ki} X_{li} + e_i \tag{1}$$

$$\boldsymbol{\beta}^{\mathrm{T}} = [\beta_0, \beta_1, \beta_2, \ \dots, \beta_m] \tag{2}$$

The response value for the ith experimental run is represented by Y_i. In this study, five different regression equations—which belong to five responses—will be calculated in the next section. X_{ki} and X_{ki}^2 terms are the linear and quadratic terms, respectively, while $X_{ki}X_{li}$ terms represent the interactions (X_1X_2, X_1X_3, X_2X_3). Finally, e_i is the residual error. The vector given in Eq. (2) contains the model's coefficients given in Eq. (1) and calculated as shown below [16–18]:

$$\beta = \left(X^T X\right)^{-1} \left(X^T Y\right) \tag{3}$$

Y is referred to as the response, and it is denoted by a column vector. In this study, the response values are obtained using Maxwell simulations. **X** is a matrix, and it is made up of the various combinations of the design parameters involved in the experimental design. The first column of the X is made up of 1 s for the model's constant term (β_0). The second, third, and fourth columns contain the factor values X_1, X_2, and X_3, respectively. The experiments in this study are divided into 14 runs. These three columns (the second, third, and fourth columns) and 14 rows are identical to the experimental design. The squares of X_1, X_2, and X_3 make up the 5th, 6th, and 7th columns of the **X** matrix, respectively. The same issue applies to interactions. By multiplying the related columns of X_1, X_2, and X_3, the interactions are placed in the 8th, 9th, and 10th columns of the **X** matrix.

After mathematical modeling, R^2 (coefficient of determination) is calculated to determine whether the factors are sufficient to describe the response change. To put it another way, R^2—which is presented in Eq. (4)—represents the level of explanatory power between the regression model and the factors.

$$R^2 = \frac{\beta^T X^T Y - n\bar{Y}^2}{Y^T Y - n\bar{Y}^2} \tag{4}$$

In order to use these models established in Eq. (1)–(3) during the optimization phase, R^2 must be closer to 1 (which means 100%). Then this means the factors of the mathematical models are sufficient to explain the shifts at Y, and in this case this means there is no need to add new factors to the regression model. The significance of the models must be determined in the final step before optimization. This is done using analysis of variance (ANOVA). The F-test is used in ANOVA to test the significance of a regression model. In this study, we used "p-value" technique (where the p-values of the each model are calculated using Minitab statistical analysis program). When the p-value is less than the alpha (type-I error), the model is considered significant. We set confidence level at 95% in the statistical analysis. This indicates that the type-I error==0.05 (5%).

In the second phase, the optimization algorithms will be run through these five regression models to calculate the optimum factor levels. In this study, four different classes of optimization methods (RSM, GA, PSO, and MSGO) are tried on this optimization problem. RSM is a gradient-based deterministic optimization method; however, GA, PSO, and MSGO are the meta-heuristics. Meta-heuristic algorithms can be classified into different groups such as evolutionary, swarm-based, human-based, etc.

Since its introduction in 1951, the RSM has become a commonly preferred design of experiment (DOE) approach for modeling and optimizing processes with a small number of experimental runs [16–18]. In this study, RSM is applied by using "Minitab Response optimizer Module," which uses gradient search algorithm in its background.

2.2 Genetic algorithm (GA)

Meta-heuristic algorithms are stochastic optimization methods that are heavily influenced by nature. In 1975, Holand created GA, a search and optimization technique [18]. Natural selection and genetic concepts are used to replicate the evolutionary process in nature. It operates based on probability laws and simply requires the purpose function. The solution area is partially investigated by GA, resulting in a more efficient search in a shorter amount of time. Chromosomes are created in the initial phase of GA to explore potential solutions. Chromosome set represents the generation's population. Selection, crossover, and mutation are the three GA operators. These operators drive the evolution of chromosomes in a generation toward the following generation. There are several uses for GA, including scheduling, vehicle routing, and transportation. GA is an evolutionary-based algorithm [19, 20]. According to Haupt and Haupt [21], since the chromosomes are not decoded before calculating the cost function, continuous GA is faster than binary GA. As a result, instead of binary GA, continuous GA was used in this study because it has the advantage of requiring less storage. In this study, the crossing method, in which Haupt & Haupt [19] combine the extrapolation method with a crossing method, is used.

2.3 Particle swarm optimization (PSO) algorithm

Particle swarm optimization (PSO) algorithm is invented by Kennedy & Eberhart [22] in 1995, and it is the first swarm-based meta-heuristic algorithm. Every possible solution in PSO is represented by a particle. The distances between a particle's present position and its best position and the best position of the group are used in PSO to update a particle's velocity [23–25].

The velocity vector and position vector for the *i*th particle are shown by v_i and x_i, respectively, in the D-dimensional search space (where $v_i = (v_{i1}, v_{i2}, \dots, v_{iD})$ and $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$). After random initialization of particles, each particle's velocity and position are updated as specified in Eq. (5) and Eq. (6) [25].

$$\mathrm{v_i(t+1) = wv_i(t) + c_1r_1(p_i - x_i(t)) + c_2r_2\left(p_g - x_i(t)\right)} \tag{5}$$

$$\mathrm{x_i(t+1) = x_i(t) + v_i(t+1)} \tag{6}$$

In these equations, w stands for the inertia weight and is used to regulate how the previous velocity affects the new. The best past positions of the ith individual and all particles in the current generation are represented, respectively, by p_i and p_g. The constants c_1 and c_2 are used to weight the positions. The uniformly distributed values between [0, 1] are $[r_1]$ and $[r_2]$. **Figure 1** shows the algorithm's progress [25].

2.4 The modified social group optimization (MSGO) algorithm

MSGO is a human-based optimization algorithm and invented in 2020. It is proposed by Naik et al. [26] by improving the acquiring phase of social group optimization (SGO) algorithm [27] and introducing a self-awareness probability factor. It is based on an individual's social behavior in a group to solve complex problems.

In MSGO, each member of the group (person) stands in for a potential solution, and the human traits—which stand in for a person's dimension—represent the amount of design variables in the issue. **Figure 2** below shows the pseudocode for the improvement phase. In **Figure 2**, P_i represents the members of the social group made up of N individuals, where $i = 1,2,3,\dots,N$. Each individual additionally has D traits ($P_i = (P_{i1}, P_{i2}, \dots, P_{iD})$). The self-introspection parameter between [0,1] and $rand \sim U(0, 1)$ is called c. The best member of the group is *gbest*, who works to spread knowledge among all people. *Gbest* will then be able to assist the group as a whole in learning more. Eq. (7) presents the aim as a minimization problem. **Figure 2** and Eq. (7) illustrate the update for each individual [26, 27].

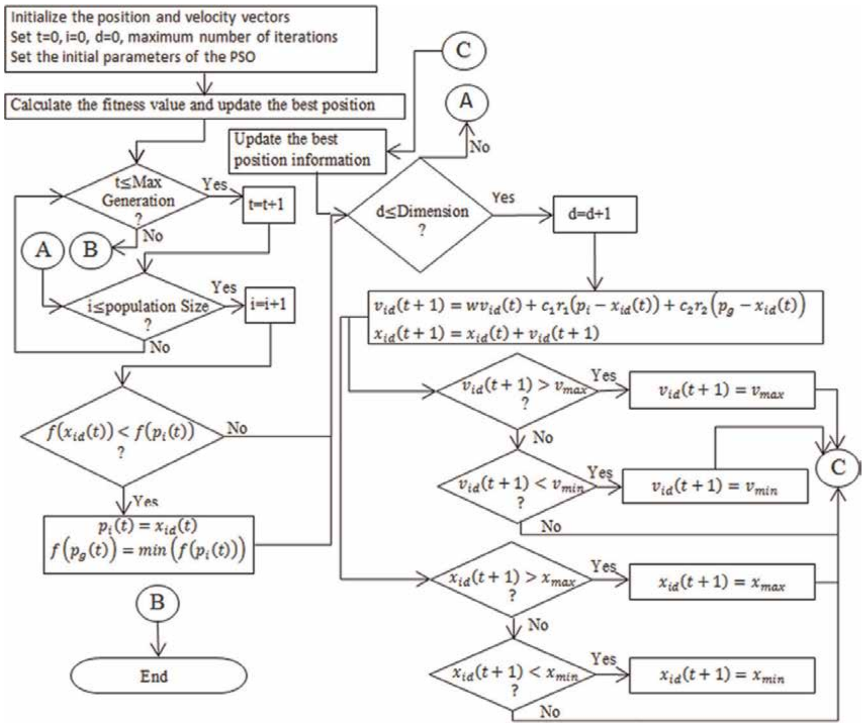


Figure 1.
The PSO algorithm.

For i=1:N
    For j=1:D
        $Pnew_{ij} = c * P_{ij} + rand * (gbest(j) - P_{ij})$
    End For
End For
Accept *Pnew* if it gives a better fitness

Figure 2.
The improving phase.

$$[minvalue, index] = min\{f(P_i), i = 1, 2, \ldots, N\} andgbest = P(index, :) \quad (7)$$

A person interacts with the group's best member (*gbest*) as well as other group members at random during the learning phase in order to gain information. In other terms, gbest is the best member of the group. A person learns something new if someone else is more knowledgeable. The person with the most knowledge, or "gbest," has the most influence over others to learn from. Even if they are more knowledgeable than they are, group members can teach a person something new. The acquiring phase is represented by Eq. (8) and **Figure 3** [26, 27].

$$[minvalue, index] = min\{f(P_i), i = 1, 2, \ldots, N\} \text{ and } gbest = P(index, :) \quad (8)$$

For i=1:N
Select randomly a person P_r (where $i \neq r$)
    If $f(P_i) < f(P_r)$
        For j=1:D
            $Pnew_{ij} = P_{ij} + rand_1 * (P_{ij} - P_{rj}) + rand_2 * (gbest(j) - P_{ij})$
        End For
    Else
        For j=1:D
            $Pnew_{i,:} = P_{i,:} + rand_1 * (P_{r,:} - P_{i,:}) + rand_2 * (gbest(j) - P_{ij})$
        End For
    End If
Accept $Pnew$ if it gives a better fitness than P
End For

Figure 3.
The acquiring phase for SGO.

For i=1:N
Select randomly a person P_r (where $i \neq r$)
    If $f(P_i) < f(P_r)$
        If rand > SAP
            For j=1:D
                $Pnew_{ij} = P_{ij} + rand_1 * (P_{ij} - P_{rj}) + rand_2 * (best_P(j) - P_{ij})$
            End For
        Else
            For j=1:D
                $Pnew_{i,:} = lb + rand * (ub - lb)$
            End For
        End If
    Else
        For j=1:D
            $Pnew_{ij} = P_{ij} + rand_1 * (P_{rj} - P_{ij}) + rand_2 * (best_P(j) - P_{ij})$
        End For
    End If
Accept $Pnew$ if it gives a better fitness than P
End For

Figure 4.
The acquiring phase for MSGO.

where the updated values at the conclusion of the improving phase are P_i values.

By changing the acquisition step of the SGO algorithm, the MSGO algorithm was created. The improving phase, however, is identical to SGO. Each social group member is still interacting with the finest individual throughout this period (best_p). Each person also engages in interaction with the other group members to learn. During this stage, if the other person knows more, the person learns something new. If one person knows more than another and that person has a greater self-awareness probability (SAP) of learning that knowledge, then that person learns something new from that other person. SAP is the capacity to learn from others, according to its definition. Modified acquisition phase is shown in Eq. (9), and **Figure 4** below shows a minimization problem [26, 27]:

$$[\text{value}, \text{index\_num}] = \min\{f(P_i), i = 1, 2, \ldots, N\} \text{ and } \text{best}_P = P(\text{index\_num}, :) \quad (9)$$

The relevant design variable's upper and lower bounds are shown in **Figure 4** as *lb* and *ub*, respectively. It is proposed to select the SAP between: $0.6 \leq SAP \leq 0.9$. According to the literature, MSGO shows best performance for SAP = 0.7 and c = 0.2 [26, 27].

3. Experimental results and discussions

We used an 18-poled 8000 rpm 7 kVA PMSG in this study. The design is done using Maxwell. **Table 2** lists the design parameters. The PMSG's structure is also shown in **Figure 5**. The rated power factor of the PMSG is 1.0. All of the winding material in the Maxwell design is standard copper. Lamination is done with Si-Fe. Finally, the insulation material H-Class is chosen.

The goal of the first stage is to use regression modeling to find the mathematical relationship between the factors (magnet thickness (X_1), offset (X_2), and embrace (X_3)) and the responses (efficiency (%), rated torque (N.m), air-gap flux density (Tesla), armature current density (A/mm^2), armature thermal load (A^2/mm^3)) by using regression modeling. "RSM face-centered design" is used to create an experiment to complete this phase. The factor levels for this experimental design are shown in **Table 3**. **Figure 6** shows a graphical representation of the experimental design. The level-2 for embrace in a standard face-centered design is 0.8%. However, due to the restrictions of the serially configured production line, we used 0.8% in the experimental design instead of 0.75%.

According to the graphical representation of experimental design that is presented in **Figure 6**, it can be clearly indicated that there are 15 experimental runs for three factors. However, no simulation could be made in Ansys Maxwell for the (−1, +1, +1)

Name	Value	Unit	Part	Description
Machine type	N/A	—	—	3-phase adjust speed PMSG
Inner dia.	100	Mm	Stator	Gap side core diameter
Outer dia.	160	Mm	Stator	Yoke side core diameter
Length	50	Mm	Stator	Length of the core
Skew width	1	Units	Stator	Slot range number
Slot type	3	N/A	Stator	Circular
Slots	54	Units	Stator	Number of slots
Bs1	2.7	Mm	Stator	Tooth width
Hs0	0.5	Mm	Stator	Slot opening height
Hs2	23	Mm	Stator	Slot height
Inner dia.	30	Mm	Rotor	Gap side core diameter
Outer dia.	99	Mm	Rotor	Yoke side core diameter
Length	50	Mm	Rotor	Length of the core
Poles	18	—	Rotor	Number of poles
Magnet	NdFe35	—	Rotor	Magnet type

Table 2.
Design parameters of 18-poled 8000 rpm 7 kVA PMSG.

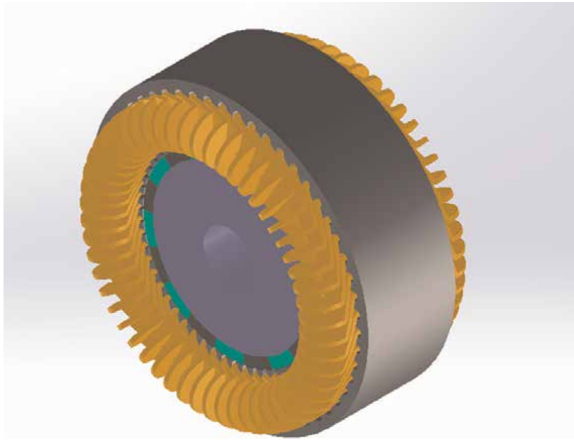


Figure 5.
Structure of the PMSG.

Factors	Symbols	Unit	Levels		
			−1	0	1
Magnet thickness	X_1	mm	2	4	6
Offset	X_2	mm	0	20	40
Embrace	X_3	%	0.5	0.8	1

Table 3.
Levels of factors.

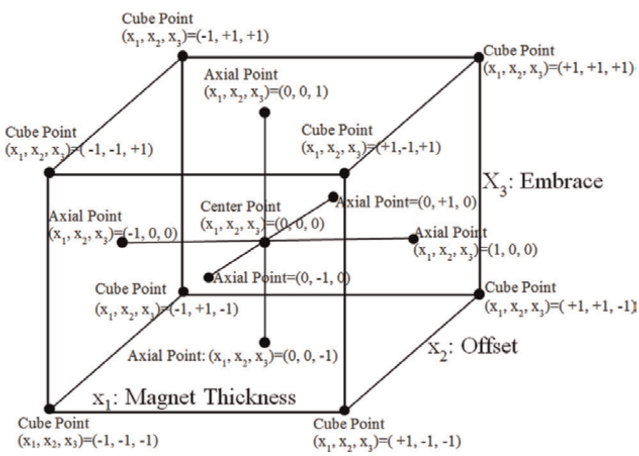


Figure 6.
Graphical representation of RSM face-centered design.

experiment (magnet thickness: 2 mm, offset: 40 mm, embrace: 1%) in this experimental design. It is not a suitable design because the offset magnet is much larger than the thickness. Therefore, since such a magnet cannot be produced, this simulation does not yield results. At the end of Maxwell simulation, it gives an error “ARC Offset is too big.” The findings of 14 experimental runs using Maxwell simulations are

Run i	Factors						Responses				
	Uncoded			Coded			Efficiency (%)	Rated Torque (N.m)	Air-Gap Flux Density (T)	Armature Current Density (A/mm^2)	Armature Thermal Load (A^2/mm^3)
	X_{i1}	X_{i2}	X_{i3}	X_{i1}	X_{i2}	X_{i3}	Y_{i1}	Y_{i2}	Y_{i3}	Y_{i4}	Y_{i5}
1	2	0	0.5	−1	−1	−1	94.08	8.88	0.89	9.86	726.16
2	6	0	0.5	1	−1	−1	93.36	8.95	1.01	10.50	822.86
3	2	40	0.5	−1	1	−1	92.58	9.02	0.89	11.17	931.11
4	6	40	0.5	1	1	−1	92.21	9.06	1.02	11.80	982.58
5	2	0	1	−1	−1	1	97.54	8.56	0.89	6.11	278.45
6	6	0	1	1	−1	1	98.02	8.52	1.02	5.42	219.37
7	6	40	1	1	1	1	94.69	8.82	1.02	9.30	645.47
8	2	20	0.8	-1	0	0	96.61	8.64	0.89	7.28	396.00
9	6	20	0.8	1	0	0	97.00	8.61	1.02	6.78	343.20
10	4	0	0.8	0	-1	0	97.46	8.57	0.99	6.22	288.41
11	4	40	0.8	0	1	0	95.01	8.79	0.99	8.99	602.48
12	4	20	0.5	0	0	-1	93.61	8.92	0.99	10.29	789.81
13	4	20	1	0	0	1	97.53	8.57	0.99	6.13	280.17
14	4	20	0.8	0	0	0	97.07	8.61	0.99	6.72	336.90

Table 4.
Ansys Maxwell simulation results.

presented in **Table 4**. The disadvantage of making genuine PMSG prototypes—which is unpredictable due to expenses—is avoided in this approach. The original uncoded factor levels are also coded by using Eq. (10) given below. The mathematical modeling will be performed in terms of both uncoded and coded factor levels. Mathematical models for uncoded factor levels display the real relationship to the readers, while the models for coded factor levels will be used in the optimization phase (the details of which will be expanded in the following paragraphs).

$$X_{coded} = \frac{X_{uncoded} - ((X_{max} + X_{min})/2)}{(X_{max} - X_{min})/2} \quad (10)$$

Minitab program is used for regression modeling and significance tests. The mathematical models for the uncoded factor levels are given in Eqs. (11) and (15). **Table 5** shows the R^2 statistics associated with the regression models.

	Y_1	Y_2	Y_3	Y_4	Y_5
R^2 (%)	99.35	99.57	100	99.25	99.39
R^2 (prediction) (%)	88.24	93.14	99.7	85.07	89.32
R^2 (adjusted) (%)	97.88	98.59	99.99	97.57	98.03

Table 5.
Summary of coefficient of determination values.

$$\begin{aligned}\hat{Y}_1 = {} & 81.0151784520298 + 0.0227200449501287X_1 + 0.090808386009271X_2 \\ & + 34.4411803062228X_3 - 0.0526452802359875X_1^2 - 0.00195145280235988X_2^2 \\ & - 18.6931720747296X_3^2 + 0.000938938053097341X_1X_2 \\ & + 0.513403216743922X_1X_3 - 0.0987609565950274X_2X_3\end{aligned} \tag{11}$$

$$\begin{aligned}\hat{Y}_2 = {} & 10.0924651495997 + 0.00532434330664408X_1 - 0.00804942407641527X_2 \\ & - 3.24443902233461X_3 + 0.00413938053097343X_1^2 + 0.000178893805309734X_2^2 \\ & + 1.77530383480827X_3^2 - 0.0000876106194690151X_1X_2 \\ & - 0.0481433487849417X_1X_3 + 0.00866354122770053X_2X_3\end{aligned} \tag{12}$$

$$\begin{aligned}\hat{Y}_3 = {} & 0.717642281219273 + 0.0998102261553588X_1 + 0.000230039331366765X_2 \\ & + 0.0165307767944939X_3 - 0.00900958702064896X_1^2 - 0.0000025959X_2^2 \\ & - 0.0168692232055067X_3^2 + 0.0000584070796460183X_1X_2 \\ & + 0.00464847590953787X_1X_3 - 0.000464847590953787X_2X_3\end{aligned} \tag{13}$$

$$\begin{aligned}\hat{Y}_4 = {} & 22.825129006883 - 0.0399906588003885X_1 - 0.112362143559489X_2 \\ & -33.6265606686333X_3 + 0.0581474926253684X_1^2 + 0.00201897492625369X_2^2 \\ & +17.8853726647001X_3^2 + 0.00091681415929201X_1X_2 - 0.603758603736482X_1X_3 \\ & +0.121209193706982X_2X_3\end{aligned} \tag{14}$$

$$\begin{aligned}\hat{Y}_5 = {} & 2449.75496212952 - 0.949312895068736X_1 - 11.2799654586319X_2 \\ & - 4567.06018645878X_3 + 6.74597935103238X_1^2 + 0.257072293510325X_2^2 \\ & + 2496.87546116028X_3^2 - 0.130220353982304X_1X_2 - 67.5995269700804X_1X_3 \\ & + 12.0458653954207X_2X_3\end{aligned} \tag{15}$$

The R^2 values presented in **Table 5** are very close to 100%—which means the selected design parameters (magnet thickness, offset, embrace) are sufficient to mathematically model the responses. ANOVA is used to determine the model's significance. For this purpose, P-value approach is used. The summary for the ANOVA results is presented in **Table 6**.

	Y_1	Y_2	Y_3	Y_4	Y_5
P-Value	0.001	0.000	0.000	0.001	0.000
Test	<0.05	<0.05	<0.05	<0.05	<0.05
Result	Significant	Significant	Significant	Significant	Significant

Table 6.
Summary of ANOVA results.

ANOVA results presented in **Table 6** indicate that all the calculated p-values are less than α = 0.05 (5%)—which means each mathematical model is significant and can be used in optimization phase. The RSM face-centered design looks to accurately reflect the supplied set of alternator design parameters. **Table 7** displays the prediction performances of the mathematical models. $\hat{Y}_i$ is the Minitab predictions (expected values) while Y_i is the simulation results obtained from Maxwell (observed values). The prediction error percentage is denoted by PE(%) and computed using Eq. (16):

$$PE_i(\%) = \frac{|Y_i - \hat{Y}_i|}{\hat{Y}_i}100 \tag{16}$$

Results provided in **Table 7** show that the regression models good fit the observed values and the PE(%) is quite low. Also the confirmation tests are performed for the mathematical models. For this purpose a new dataset that is composed of five new Maxwell simulation results—which is not used in the mathematical modeling phase previously—is used. Confirmations are presented in **Table 8**.

According to the confirmation results indicated in **Table 8**, the overall PE(%) is acceptable. The comparisons shown in **Tables 7** and **8** indicate that these numerical models can be used for optimization.

In the optimization phase, four different optimization methods (RSM, GA, PSO, and MSGO) from four different classes are tested for calculating the optimum design parameters. To establish the optimum factor levels, the optimization

Run (i)	Efficiency (%)			Rated torque (N.m)			Air-gap flux density (Tesla)			Armature current density (A/mm^2)			Armature thermal load (A^2/mm^3)		
	Y_{i1}	$\hat{Y}_{i1}$	PE_{i1} (%)	Y_{i2}	$\hat{Y}_{i2}$	PE_{i2} (%)	Y_{i3}	$\hat{Y}_{i3}$	PE_{i3} (%)	Y_{i4}	$\hat{Y}_{i4}$	PE_{i4} (%)	Y_{i5}	$\hat{Y}_{i5}$	PE_{i5} (%)
1	94.08	93.911	0.18	8.88	8.893	0.15	0.89	0.890	0.01	9.86	10.032	1.71	726.16	747.929	2.91
2	93.36	93.344	0.02	8.95	8.951	0.01	1.01	1.010	0.02	10.50	10.525	0.24	822.86	824.805	0.24
3	92.58	92.521	0.06	9.02	9.024	0.04	0.89	0.890	0.04	11.17	11.265	0.85	931.11	938.546	0.79
4	92.21	92.104	0.12	9.06	9.067	0.08	1.02	1.020	0.01	11.80	11.905	0.89	982.58	994.586	1.21
5	97.54	97.625	0.09	8.56	8.554	0.07	0.89	0.890	0.02	6.11	6.029	1.34	278.45	269.456	3.34
6	98.02	98.085	0.07	8.52	8.515	0.05	1.02	1.020	0.03	5.42	5.315	1.98	219.37	211.132	3.90
7	94.69	94.870	0.19	8.82	8.805	0.17	1.02	1.020	0.02	9.30	9.119	1.98	645.47	621.831	3.80
8	96.61	96.754	0.15	8.64	8.629	0.13	0.89	0.890	0.05	7.28	7.093	2.63	396.00	375.788	5.38
9	97.00	96.878	0.13	8.61	8.622	0.14	1.02	1.020	0.00	6.78	6.936	2.24	343.20	361.126	4.96
10	97.46	97.496	0.04	8.57	8.567	0.04	0.99	0.990	0.00	6.22	6.209	0.18	288.41	281.927	2.30
11	95.01	94.996	0.01	8.79	8.794	0.05	0.99	0.990	0.05	8.99	8.970	0.22	602.48	606.677	0.69
12	93.61	93.961	0.37	8.92	8.896	0.28	0.99	0.990	0.03	10.29	9.892	4.02	789.81	746.654	5.78
13	97.53	97.201	0.34	8.57	8.595	0.29	0.99	0.990	0.01	6.13	6.497	5.65	280.17	321.040	12.73
14	97.07	97.026	0.04	8.61	8.609	0.01	0.99	0.991	0.08	6.72	6.782	0.91	336.90	341.473	1.34

Table 7.
Regression model performances.

Run (i)	Efficiency (%)			Rated torque (N.m)			Air-gap flux density (Tesla)			Armature current density (A/mm^2)			Armature thermal load (A^2/mm^3)		
	Y_{i1}	$\hat{Y}_{i1}$	PE_{i1} (%)	Y_{i2}	$\hat{Y}_{i2}$	PE_{i2} (%)	Y_{i3}	$\hat{Y}_{i3}$	PE_{i3} (%)	Y_{i4}	$\hat{Y}_{i4}$	PE_{i4} (%)	Y_{i5}	$\hat{Y}_{i5}$	PE_{i5} (%)
15	96.75	96.237	0.53	8.63	8.679	0.57	0.96	0.947	1.37	7.11	7.623	6.73	377.61	443.702	14.90
16	95.33	95.455	0.13	8.76	8.756	0.05	1.01	1.011	0.14	8.67	8.382	3.44	560.40	548.170	2.23
17	97.73	97.850	0.12	8.55	8.535	0.18	1.01	1.014	0.43	5.84	5.800	0.69	254.69	236.384	7.74
18	94.82	94.619	0.21	8.81	8.833	0.26	1.01	1.014	0.41	9.18	9.296	1.25	628.25	659.757	4.78
19	96.89	96.421	0.49	8.62	8.664	0.51	1.01	1.014	0.43	6.95	7.442	6.61	360.79	420.246	14.15

Table 8.
Confirmation tests.

algorithms will be run through these five regression models. RSM is a gradient-based method, while GA (evolutionary-based algorithm), PSO (swarm intelligence-based algorithm), and MSGO (human-based algorithm) are meta-heuristic optimization methods [24]. In this study, the performance of the multiobjective optimization using meta-heuristics is done by combining all the responses in one objective function independent from their units. To do this, the response functions must be recalculated by using the coded factor levels (instead of original levels) between −1 (for minimum value for the factor level) and + 1 (for maximum value for the factor level). The regression models calculated from coded factor levels are given in Eqs. (17)–(21):

$$\begin{aligned}\hat{Y}_{1,coded} = {} & 96.7492070304818 + 0.0107779533642339X_1 - 1.15129382638011X_2 \\ & + 1.61995398230089X_3 - 0.210581120943952X_1^2 - 0.780581120943952X_2^2 \\ & - 1.1683232546706X_3^2 + 0.0375575221238924X_1X_2 \\ & + 0.25670160837196X_1X_3 - 0.493804782975135X_2X_3\end{aligned} \tag{17}$$

$$\begin{aligned}\hat{Y}_{2,coded} = {} & 8.63435501474926 + 0.0011593271526898X_1 + 0.105070831577469X_2 \\ & - 0.150196460176991X_3 + 0.0165575221238939X_1^2 \\ & + 0.0715575221238936X_2^2 + 0.110956489675516X_3^2 \\ & - 0.00350442477876052X_1X_2 - 0.0240716743924708X_1X_3 \\ & + 0.0433177061385025X_2X_3\end{aligned} \tag{18}$$

$$\begin{aligned}\hat{Y}_{3,coded} = {} & 0.990846656833825 + 0.0647760570304818X_1 \\ & + 0.000223942969518185X_2 + 0.000130973451327435X_3 \\ & - 0.0360383480825959X_1^2 - 0.00103834808259587X_2^2 \\ & - 0.00105432645034415X_3^2 + 0.0023362831858407{1}X_1X_2 \\ & + 0.00232423795476892X_1X_3 - 0.00232423795476893X_2X_3\end{aligned} \tag{19}$$

$$\hat{Y}_{4,coded} = 7.07668220255654 - 0.0185867748279251X_1 + 1.25942010816126X_2 - 1.69733805309735X_3 + 0.232589970501475X_1^2 + 0.807589970501475X_2^2 + 1.11783579154375X_3^2 + 0.0366725663716815X_1X_2 - 0.30187930186824X_1X_3 + 0.606045968534907X_2X_3 \quad (20)$$

$$\hat{Y}_{5,coded} = 377.792067158309 - 0.571060788032038X_1 + 150.328878248349X_2 - 212.806948672566X_3 + 26.9839174041298X_1^2 + 102.82891740413X_2^2 + 156.054716322517X_3^2 - 5.20881415929203X_1X_2 - 33.7997634850401X_1X_3 + 60.2293269771036X_2X_3 \quad (21)$$

The objective function is given in Eq. (22) and Eq. (23). The aim is to maximize the efficiency (Y_1), while holding the air-gap flux density (Y_3) at 1 Tesla and minimizing the rest of the responses (Y_2, Y_4, Y_5).

$$Z = |(Y_{1,coded}/\max(Y_{i1}))| - |(Y_{2,coded}/\max(Y_{i2}))| - \left|\left(Y_{3,target}/\max(Y_{i3})\right) - (Y_{3,coded}/\max(Y_{i3}))\right| - |(Y_{4,coded}/\max(Y_{i4}))| - |(Y_{5,coded}/\max(Y_{i5}))| \quad (22)$$

Min Z s.t. X1 $\in[-1,1]$; X2$\in[-1,1]$; X3$\in[-1,1]$

Note that the $Y_{3,target} = 1$ Tesla in the equation of Z. In addition; $\max(Y_{i1})$, $\max(Y_{i2})$, $\max(Y_{i3})$, $\max(Y_{i4})$, and $\max(Y_{i5})$ are the maximum observed response values presented in **Table 4** (which are 98.02, 9.06, 1.02, 11.8, and 982.58 for this problem, respectively). If the readers would like to use the Matlab codes referred in the reference [28] for MSGO, note that the signs of the each term are the exact opposite (since the codes in the reference are coded according to maximization problems). Then the *Z* function set in the Matlab code is given in Eq. (23):

$$Z = -|(Y_{1,coded}/98.02)| + |(Y_{2,coded}/9.06)| + |(1/1.02) - (Y_{3,coded}/1.02)| + |(Y_{4,coded}/11.8)| + |(Y_{5,coded}/982.58)| \quad (23)$$

MSGO, PSO, GA, and RSM are run through these mathematical models to perform multi-objective optimization. **Table 9** summarizes the optimized factor levels and the calculated CPU times (at a PC: Intel i5 4GB RAM), for each method. In this table, *nPop* and *MaxIt* represent the population size and maximum number of iterations, respectively. For MSGO, c and SAP are set as 0.2 and 0.7, respectively. In PSO, the parameters of the algorithm are set as: w = 1, w_{damp} = 0.99, c_1 = 1.5, c_2 = 2.0. In the GA, we use the crossover rate = 0.50 and the mutation rate = 0.20. The optimization results for these optimization methods are presented in **Table 10**.

Results presented in **Table 10** indicate that the meta-heuristics superiors RSM with a quite bit difference. When compared among themselves, MSGO and PSO together give better results than GA. The MSGO and PSO give the same optimization results. So

	Run parameters		Coded factor levels			Uncoded factor levels			CPU time
Method	***nPop***	***MaxIt***	X_1	X_2	X_3	X_1	X_2	X_3	
MSGO	30	2000	**0.7396**	**-1**	**1**	**5.48**	**0**	**1**	5
PSO	100	1000	**0.7396**	**-1**	**1**	**5.48**	**0**	**1**	7
RSM	N/A	N/A	0.2	−0.5	1	4.4	10	1	N/A
GA	8	100,000	0.1528	−1	0.9485	4.31	0	0.99	9

Table 9.
Optimized factor levels for each method.

	$\hat{Y}_1$	$\hat{Y}_2$	$\hat{Y}_3$	$\hat{Y}_4$	$\hat{Y}_5$
Target:	Max	Min	1 Tesla	Min	Min
MSGO	**98.1202**	**8.513**	1.0192	**5.3024**	**206.5071**
PSO	**98.1202**	**8.513**	1.0192	**5.3024**	**206.5071**
RSM	97.8696	8.5352	1.0025	5.7079	236.194
GA	98.0959	8.5157	**1.0002**	5.3895	209.4387

Table 10.
Summary of the optimization results.

the optimized factor levels of MSGO and also same as PSO are used for designing the optimized design. The optimum factor levels are calculated as: magnet thickness: 5.48 mm, offset: 0 mm, embrace: 1%. However, confirmation results of these four methods are given together in **Table 11**. The observed responses and the fitted responses are presented in **Table 11**. Also the PE (%) values are calculated for each response in terms of the methods.

The optimized PMSG's magnetic flux distribution and the voltage graphs are presented in **Figures 7** and **8**, respectively. The graphs for flux linkage, power, and torque are given in **Figures 9**, **10**, and **11**, respectively.

In order to obtain the desired responses, the embrace must be maximum. Also the magnet thickness must be bigger than 4 mm. For this sample PMSG structure in the article, the results showed that offset has no discernible effect on responses (causes little changes).

As previously stated, this issue is only relevant to the PMSG in this case study. Additional optimization methods can be used to expand on these findings and discussions such as bat algorithm (BA) [29], grew wolf optimizer (GWO) [30], whale

Method	Y_{i1}	$\hat{Y}_{i1}$	PE_{i1} (%)	Y_{i2}	$\hat{Y}_{i2}$	PE_{i2} (%)	Y_{i3}	$\hat{Y}_{i3}$	PE_{i3} (%)	Y_{i4}	$\hat{Y}_{i4}$	PE_{i4} (%)	Y_{i5}	$\hat{Y}_{i5}$	PE_{i5} (%)
MSGO	**98.03**	**98.1202**	0.09	**8.52**	**8.513**	0.08	1.01	1.0192	0.90	**5.4**	**5.3024**	1.84	**217.9**	**206.5071**	5.52
PSO	**98.03**	**98.1202**	0.09	**8.52**	**8.513**	0.08	1.01	1.0192	0.90	**5.4**	**5.3024**	1.84	**217.9**	**206.5071**	5.52
RSM	97.84	97.8696	0.03	8.54	8.5352	0.06	**1**	**1.0025**	0.25	5.69	5.7079	0.31	241.3	236.194	2.16
GA	98.01	98.0959	0.09	**8.52**	**8.5157**	0.05	**1**	**1.0002**	0.02	5.42	5.3895	0.57	219.76	209.4387	4.93

Table 11.
Confirmations for the optimized factor levels.

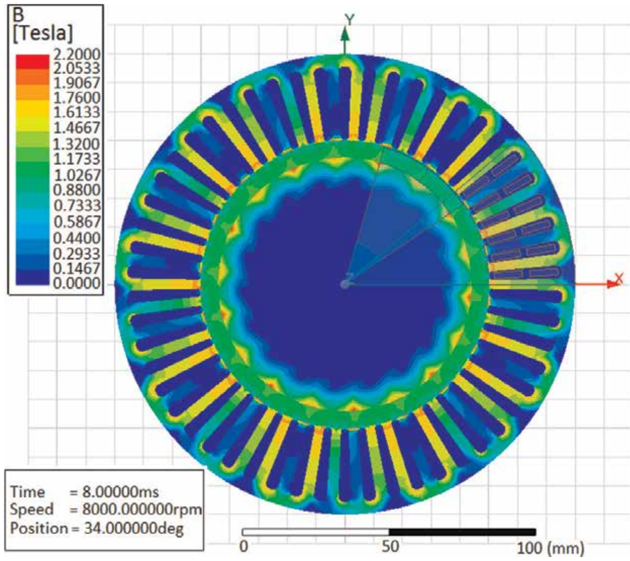


Figure 7.
Magnetic flux density distribution of the optimized PMSG.

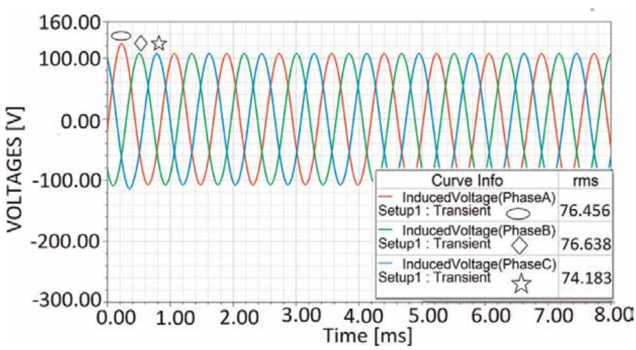


Figure 8.
Voltage graph of the optimized PMSG.

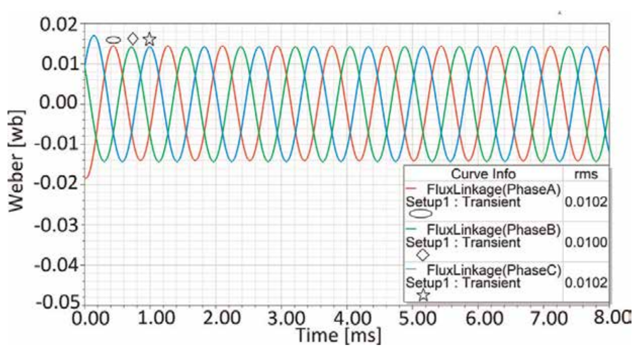


Figure 9.
Flux linkage of the optimized PMSG.

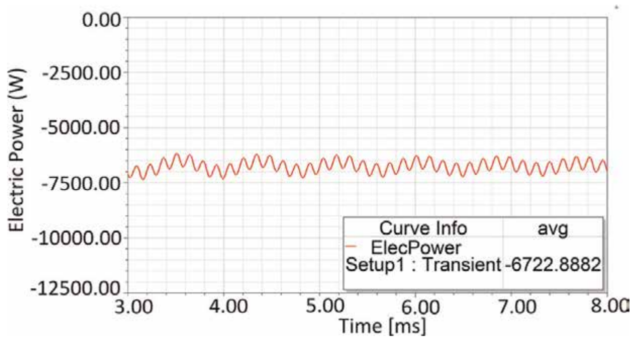


Figure 10.
Power graph of the optimized PMSG.

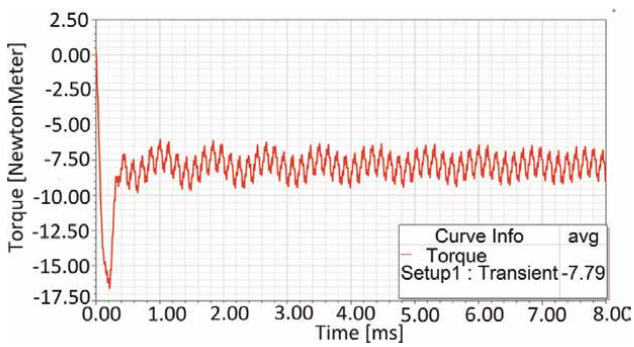


Figure 11.
Torque graph of the optimized PMSG.

optimization (WOA) [31], grasshopper optimization algorithm (GOA) [32] etc., in the future research studies.

4. Conclusion

The design of an 18-poled 8000 rpm 7 kVA PMSG is optimized in this study. The goal is to determine the optimal levels of magnet thickness (MH), offset, and embrace (EMB) to keep the air-gap flux density at 1 tesla while maximizing efficiency and minimizing other responses. For this purpose, Ansys Maxwell is used for calculating the responses and Minitab is used for mathematically modeling the relations between the factors and the responses by using simulation results. Then MSGO, PSO, RSM, and GA are used for optimization by running these algorithms through the regression models. Matlab coding is performed for this stage. Although the results of the four methods are nearly identical, MSGO and PSO outperform the other methods for the sample PMSG presented in this study. Although the results of the four methods are nearly identical, one advantage of RSM is that it does not require program coding and allows for visual examination of the relationships between factors and responses. The RSM is clearly less complex than the PSO, MSGO, and GA, according to the time complexity analysis. When comparing the PSO, MSGO, and GA, it is clear that the MSGO has fewer parameters to tune and produces extremely accurate results, making it extremely efficient. In the future, we plan to expand the work to include additional design parameters, higher power groups, and additional optimization methods. The optimum factor levels are calculated as: magnet thickness: 5.48 mm, offset: 0 mm, embrace: 1% at the end of optimization phase. The embrace must be at its peak in order to obtain the desired responses. In addition, the magnet thickness must be greater than 4 mm. The results demonstrated that offset has no discernible effect on the selected responses for the selected PMSG structure in this manuscript.

Acknowledgements

We'd like to express our gratitude to Isbir Electric Company's Department of Research and Development for allowing us to use its facilities and applications.

Conflict of interest

"The authors declare no conflict of interest."

Author details

Aslan Deniz Karaoglan[1]*, Deniz Perin[2] and Kemal Yilmaz[2]

1 Department of Industrial Engineering, Balikesir University, Balikesir, Turkey

2 Department of R&D, ISBIR Electric Company, Balikesir, Turkey

*Address all correspondence to: deniz@balikesir.edu.tr

References

[1] Gizolme O, Thollon F, Clerc G, Rojat G. Shape optimization of synchronous machine rotor. International Journal of Applied Electromagnetics and Mechanics. 1998;**9**: 263-275

[2] Gillon F, Brochet P. Screening and response surface method applied to the numerical optimization of electromagnetic devices. IEEE Transactions on Magnetics. 2000;**36**: 1163-1167

[3] Jolly L, Jabbar MA, Qinghua L. Design optimization of permanent magnet motors using response surface methodology and genetic algorithms. IEEE Transactions on Magnetics. 2005; **41**:3928-3930

[4] Fang L, Jung JW, Hong JP, Lee JH. Study on high-efficiency performance in interior permanent-magnet synchronous motor with double-layer PM design. IEEE Transactions on Magnetics. 2008; **44**:4393-4396

[5] Hasanien HM, Muyeen SM. A Taguchi approach for optimum design of proportional-integral controllers in cascaded control scheme. IEEE Transaction on Power Systems. 2013;**28**: 1636-1644

[6] Zhang CJ, Chen ZH, Mei QX, Duan JJ. Application of particle swarm optimization combined with response surface methodology to transverse flux permanent magnet motor optimization. IEEE Transactions on Magnetics. 2017; **53**:8113107

[7] Chai W, Lipo TA, Kwon BI. Design and optimization of a novel wound field synchronous machine for torque performance enhancement. Energies. 2018;**11**:2111

[8] Islam MJ, Moghaddam RR. Loss reduction in a salient pole synchronous machine due to magnetic slot wedge and semi-closed stator slots. In: 13th International Conference on Electrical Machines (ICEM), 03–06 Sept. 2018. Alexandroupoli, Greece: IEEE; 2018. pp. 1267-1272

[9] Soleimani J, Ejlali A, Moradkhani M. Transverse flux permanent magnet generator design and optimization using response surface methodology applied in direct drive variable speed wind turbine system. Periodicals of Engineering and Natural Sciences. 2019;**7**:36-53

[10] Karaoglan AD, Perin D, Yilmaz K. Multiobjective design optimization of stator for synchronous generator using bat algorithm and analysis of magnetic flux density distribution. Electric Power Components and Systems. 2021;**49**: 919-929. DOI: 10.1080/15325008.2022.2049651

[11] Karaoglan AD, Perin D. Design optimization of a 4-poled 1500 rpm 25 kVA SG to obtain the desired magnetic flux density distributions by using RSM. Journal of Scientific & Industrial Research. 2022;**81**:84-93

[12] Sadeghierad M, Amini S, Ziaie S. Comparison of alternative configurations for a high speed alternator for microturbines. In: 21th International Power Systems Conference, 13–15 Nov. 2006. Tehran, Iran: PSC2006; 2006. pp. 587-595

[13] Sadeghierad M, Darabi A, Lesani H, Monsef H. Optimal design of the generator of microturbine using genetic algorithm and PSO. International Journal of Electrical Power & Energy Systems. 2010;**32**:804-808

[14] Ismagilov FR, Vavilov VE, Gusakov DV, Ou J. High-speed generator with tooth-coil winding, permanent magnets and new design of a stator magnetic core made from amorphous alloy. In: 25th International Workshop on Electric Drives-Optimization in Control of Electric Drives (IWED); 31 January-02 February 2018. Moscow, Russia: IEEE; 2018

[15] Guo J, Jin YM, Zhang Y, Xue MZ, Luan Y. Optimization design of high-speed generator for micro turbojet engine based on GDSFD-AL method. In: IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC); 08–10 May 2019. Seogwipo-si, South Korea: IEEE; 2019. pp. 144-150

[16] Montgomery DC. In: Haboken NJ, editor. Design and Analysis of Experiments. New Jersey, USA: John Wiley & Sons; 2013

[17] Mason RL, Gunst RF, Hess JL. Statistical Design and Analysis of Experiments. Haboken, NJ: John Wiley & Sons; 2003

[18] Karaoglan AD. Optimizing plastic extrusion process via grey wolf optimizer algorithm and regression analysis. Journal of Scientific & Industrial Research. 2021;**80**:34-41

[19] Holland JH. Adaptation in Natural and Artificial Systems. Cambridge, UK: MIT Press; 1975

[20] Goldberg DE. Genetic Algorithms in Search, Optimization and Machine Learning. Boston, MA: Addison-Wesley Longman Publishing Co. Inc; 1989

[21] Haupt R, Haupt SE. Practical Genetic Algorithms. Habokenm NJ: John Wiley & Sons; 2004

[22] Kennedy J, Eberhart R. Particle swarm optimization. In: IEEE International Conference on Neural Networks; 27 November – 1 December 1995. Perth, Western Australia: IEEE; 1995. pp. 1942-1948

[23] Wei Y, Qiqiang L. Survey on particle swarm optimization algorithm. Engineering Science. 2004;**5**: 87-94

[24] Lalwani S, Sharma H, Satapathy SC, Deep K, Bansal JC. A survey on parallel particle swarm optimization algorithms. Arabian Journal for Science and Engineering. 2019;**44**:2899-2923

[25] Dai H-P, Chen D-D, Zheng Z-S. Effects of random values for particle swarm optimization algorithm. Algorithms. 2018;**11**:23

[26] Naik A, Satapathy SC, Abraham A. Modified social group optimization—A meta-heuristic algorithm to solve short-term hydrothermal scheduling. Applied Soft Computing. 2020;**95**:106524

[27] Satapathy S, Naik A. Social group optimization (SGO): A new population evolutionary optimization technique. Complex & Intelligent Systems. 2016;**2**: 173-203

[28] Naik A. Modified Social Group Optimization algorithm [Internet]. 2021. Available from: https://www.mathworks.com/matlabcentral/fileexchange/78272-modified-social-group-optimization-algorithm [Accessed: January 29, 2021], MATLAB Central File Exchange.

[29] Yang XS. A New Metaheuristic Bat-Inspired Algorithm. Nature Inspired Cooperative Strategies for Optimization (NISCO 2010). Studies in Computational Intelligence. Berlin: Springer; 2010. pp. 65-74

[30] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Advances in Engineering Software. 2014;**69**:46-61

[31] Mirjalili S, Lewis A. The whale optimization algorithm. Advances in Engineering Software. 2016;**95**:51-67

[32] Saremi S, Mirjalili S, Lewis A. Grasshopper optimisation algorithm: Theory and application. Advances in Engineering Software. 2017;**105**:30-47

Chapter 9

Autonomous Role Assignment Using Contact Stimuli in Swarm Robotic Systems

Kazuaki Yamada

Abstract

This study proposes a novel autonomous role assignment method for swarm robotic systems using the response threshold model based on local interactions in a dynamic environment. Ants are social insects with high and low pheromone sensitivity. The pheromone sensitivity of ants is related to autonomous role assignment. The response threshold model was proposed to describe the pheromone sensitivity of ants. The conventional response threshold model assumes that an ant knows the number of workers in an ant colony. However, it is difficult for an ant to contact all workers because its functions are very limited. Therefore, our proposed method adopts a response threshold model based on contact stimuli with foraging ants instead of the worker ratio in an ant colony. In this study, to evaluate the proposed method's robustness in dynamic environments, we apply it to ant foraging problems in environments with varying amounts and distributions of feeds.

Keywords: swarm robotics, autonomous role assignment, local interaction, ant foraging, response threshold model

1. Introduction

Swarm robotics is an approach that applies the smart swarm behaviour [1] observed in flocks of birds, schools of fish and swarms of social insects to engineering problems [2–4]. This study focuses on the excellent functions of swarms of ants, which are social insects. Ants sustain large colonies through caste systems, with the queen at the top, which assign different roles to each caste member. The perception functions and action rules of ants are limited, and communication between them can only be conducted through different pheromones. The queen ant cannot monitor everything that happens in a colony and cannot give instructions to each ant directly. Nevertheless, ants are successfully assigned different roles, such as colony protection, food exploration and foraging, without any centralised management system [5]. The autonomous role assignment mechanism of ants may be useful for transport automation by using several autonomous mobile robots in large warehouses and for search-and-rescue operations using several autonomous drones in disaster relief.

As one of the autonomous role assignments in termites, a worker ant specialises as a soldier ant [6]. However, an appropriate role assignment system is required because an excessively increasing number of soldier ants reduce the amount of collected feeds. Therefore, the specialisation of a worker ant to a soldier ant is impeded by a soldier pheromone, the concentration of which rises with the number of soldier ants. In addition, a colony's autonomous role assignment allows it to adapt to changing circumstances. For example, when food becomes scarce, a colony needs to increase the number of ants exploring new food sources as well as the number of ants foraging for food once a new food source has been discovered. Rather than assigning roles in a top-down manner, ants assign roles appropriately through local communication using pheromones.

Bonabeau et al. [7–9] modelled this role assignment using a response threshold model. The response threshold model is an equation that describes the sensitivity of ants to pheromones. There are two types of ants [10]: one with high sensitivity to pheromones and the other with low sensitivity. These different sensitivities are thought to contribute to an autonomous role assignment. However, the conventional response threshold model uses the ratio of workers in an ant colony as an external stimulus, ignoring the crucial factor that social insects can assign roles through local communication.

In contrast, Gordon et al. [11–13] revealed that an ant's tendency to perform midden work[1] or foraging work is related to the recent history of its contact with an ant engaged in those works based on the observation of red harvester ants. Our research group has proposed an autonomous role assignment and task allocation method with local interactions in scalable swarm robotic systems [14, 15]. The method used a response threshold model using the ratio of encountered foraging ants in the short term as an external stimulus and mimicking the action rules of real ants. We applied the proposed method to ant foraging problems in a dynamic environment with a varying number of ants [15]. Through simulation results, we confirmed that, during internal environment fluctuations, the proposed method using local interactions outperformed the conventional method using global information.

In this study, we propose a simple autonomous role assignment method using contact stimuli with foraging ants, rather than the ratio of encountered foraging ants in the short term. To evaluate the proposed method's effectiveness, we apply the method to ant foraging problems in a dynamic environment with fluctuating amounts and distributions of feeds. Through simulation results, we demonstrate that, during external environment fluctuations, the proposed method using local interaction outperforms the conventional method using global information. In addition, we demonstrate that the method can successfully perform role assignment in an ant colony by switching between exploring and foraging behaviours through contact stimuli with foraging ants.

The rest of this chapter is organised as follows. Section 2 explains how to model an ant foraging problem. Section 3 shows the new response threshold model. Section 4 demonstrates the proposed method's effectiveness through simulations. Conclusions and future work are discussed in Section 5.

2. Foraging problem

This section models an ant foraging problem as a multi-agent simulation following previous studies [16]. In this model, an ant is modelled as an agent. An agent has the following three functions:

[1] Midden work is carrying objects to and sorting the refuse pile of the colony.

- **Exploring/foraging behaviour**

An agent exhibits either exploring or foraging behaviour. When an agent discovers food, the agent exhibits foraging behaviour and carries food to its nest. Otherwise, the agent explores food sources.

- **Homing/trail pheromone**

An agent possesses chemical substances termed homing and trail pheromones. The agent secretes the homing pheromone while exploring for food and the trail pheromone while carrying food to the nest. Both pheromones are volatile substances that diffuse and evaporate quickly.

- **Worker/non-worker**

An agent can be either a worker or a non-worker. While exploring food sources, an agent can perceive trail pheromones if it is a worker but cannot if it is a non-worker. While carrying food, the agent can perceive homing pheromones whether it is a worker or a non-worker.

Table 1 shows the relationship among exploring/foraging behaviour, worker/non-worker and homing/trail pheromone.

Next, we describe the modelling of perception and action. As shown in **Figure 1**, an agent can perceive the difference in pheromone level between three front cells and the current cell and can select one of three action rules before moving to the next cell. The three action rules are as follows:

	Worker	Non-worker
Exploring behaviour	able to perceive trail pheromones	unable to perceive trail pheromones
	laying homing pheromones	
Foraging behaviour	able to perceive homing pheromones	
	laying trail pheromones	

Table 1.
Relationship among behaviour, pheromones and worker/non-worker.

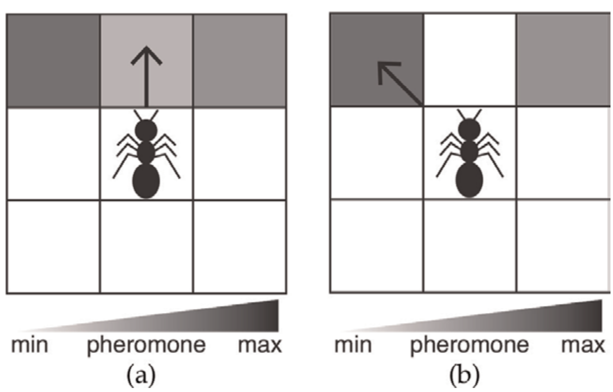


Figure 1.
Pheromone-following action rules.

	Worker	Non-worker
Exploring behaviour	Rule 2 → Rule 1	Rule 1
Foraging behaviour	Rule 3 → Rule 2 → Rule 1	

Table 2.
Relationship between worker/non-worker and behaviour rules.

1. **Random walk**

    When the difference in pheromone level is negative, an agent randomly selects one cell from three cells and moves to that cell.

2. **Pheromone trail**

    When the difference in pheromone level is positive and an agent detects the pheromone in the front cell (**Figure 1a**), the agent moves forward. On the other hand, when the agent detects pheromones in the right and left front cells, it moves to the cell with the highest pheromone level (**Figure 1b**).

3. **Turn around**

    When an agent discovers food or exhausts its homing pheromone, the agent turns to the nest. However, the agent also turns to the nest if the angle between the moving direction and the direction of the nest is greater than 90° [17].

    An agent can exhibit exploring or foraging behaviour by changing the combination of rules (**Table 2**). In exploring behaviour, an agent moves according to Rule 2 if it is a worker and can detect a trail pheromone, but an agent moves randomly according to Rule 1 if it is a non-worker. In foraging behaviour, both worker and non-worker agents follow Rule 3. If an agent detects a homing pheromone, it moves according to Rule 2. If it detects no homing pheromone, it moves according to Rule 1.

3. Response threshold model

This section proposes a new response threshold model using contact stimuli with foraging agents. Firstly, we describe the role that a response threshold model plays in autonomous role assignment and introduce the conventional response threshold model. Next, we describe the proposed response threshold model that uses contact stimuli with foraging agents as external stimuli. There are two types of ants: those sensitive to external stimuli and those insensitive to external stimuli. Sensitivity to external stimuli can be modelled using a parameter called a response threshold. An agent with a low response threshold is likely to become a worker even if its sensitivity to external stimuli is weak; however, an agent with a high response threshold is unlikely to become a worker even if its sensitivity to external stimuli is high. Thus, a response threshold can prevent outcomes in which all agents are workers or non-workers. In the conventional response threshold model, an agent changes from a worker to a non-worker with probability p and changes from a non-worker to a worker with the probability described using the following equation:

$$q = \frac{s(t)^2}{s(t)^2 + \theta(t)^2}, \tag{1}$$

where θ and s represent a response threshold and an external stimulus at time t, respectively. The response threshold is updated using Eq. (2) if the agent is a worker and using Eq. (3) if it is a non-worker. If the agent is a worker, the response threshold decreases and its sensitivity to external stimuli increases. If the agent is a non-worker, the response threshold increases and its sensitivity to external stimuli decreases.

$$\theta(t+1) = \theta(t) - \xi. \tag{2}$$

$$\theta(t+1) = \theta(t) + \psi. \tag{3}$$

In the conventional model, a stimulus s is updated by the ratio of the number of workers $N_w(t)$ to the total number of ants $N_t(t)$ in an ant colony, as described by the following equation:

$$s(t+1) = s(t) + \delta - \alpha \frac{N_w(t)}{N_t(t)}, \tag{4}$$

where δ represents an increase in loads per unit time if no ant forages. The third term on the right side of Eq. (4) represents a decrease in loads per ant to the ratio of the number of workers in the ant colony, and α represents a scale factor. That is, if the worker ratio in the ant colony decreases, the stimulus s increases with increasing loads per ant and the probability of changing from a non-worker to a worker increases. However, ants cannot know the state of all other ants. Therefore, the above equation cannot represent the mechanism by which ants can form orderly swarms through local interactions. We, therefore, propose a novel equation as follows:

$$s(t+1) = \beta cs(t), \tag{5}$$

$$cs(t) = c(t) + \gamma cs(t-1), \tag{6}$$

$$c(t) = \begin{cases} 1 & \text{if an agent contacts with a foraging agent,} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

In the proposed model, a stimulus $s(t+1)$ is updated by multiplying a contact stimulus $cs(t)$ by a scale factor β. The contact stimulus $cs(t)$ decreases by the attenuation rate γ over time if an agent does not contact a foraging agent. If an agent contacts a foraging agent, $c(t)$ is 1, otherwise 0.

4. Simulations

We applied the proposed method to an ant foraging problem and evaluated its robustness in a fluctuating external environment. The simulation results show mechanisms that can flexibly assign foraging and exploring agents in an environment with fluctuating amounts and distributions of feeds through contact stimuli with foraging agents as the local interaction.

4.1 Simulation setting

Figure 2 depicts the simulator. The simulator was constructed with reference to the following previous studies [9, 15, 16, 18], with the best simulation parameters selected through preliminary experiments. **Table 3** shows the parameters used in simulation experiments. The experimental environment comprised a two-dimensional grid space of 150 × 150 cells. The nest was placed in the centre of the environment. The simulation halted after 10,000 steps in one trial, and we conducted 50 trials in each experimentation setting. A red cell represents a worker agent, and a purple cell represents a non-worker agent. When an agent touches the feed, it carries the feed to the nest; it is represented as an orange cell. A green cell contains a trail pheromone, and a blue cell contains a homing pheromone. As the pheromone evaporates, the pheromone level decreases, and the colour of the cell becomes lighter. The homing and trail pheromones do not mix. In the initial state, food sources were randomly placed in the food source area. When the feed in one source is exhausted, the next food source is placed randomly in the area. In the simulation, we fluctuated the amount and distribution of feeds to evaluate the proposed method's robustness in a dynamic environment. The simulation alternated between three different types of environments as follows:

Type-A: In **Figure 3a**, the environment is dotted with four small food sources. Each food source includes one feed.
Type-B: In **Figure 3b**, the environment is dotted with four medium food sources. Each food source includes nine feeds.
Type-C: In **Figure 3c**, the environment has one large food source. The food source contains one hundred feeds.

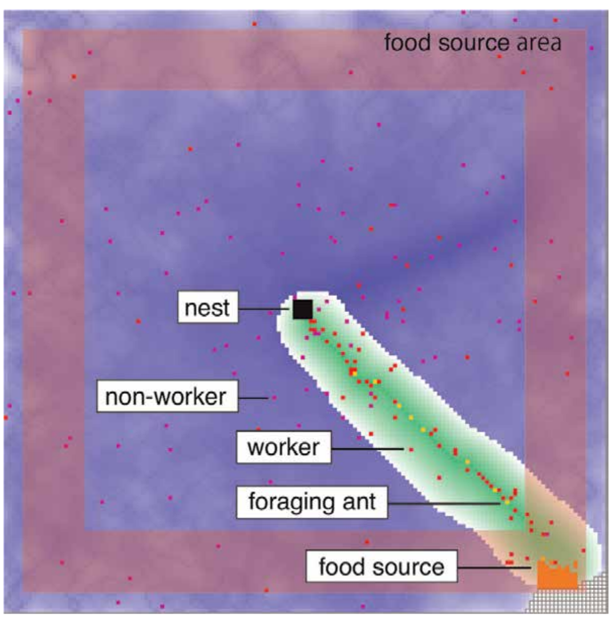


Figure 2.
Simulator.

Notation	Description	Values
Simulation		
step	the maximum number of steps in one trial	10,000
trial	the number of trials in each experiment	50
Response threshold		
p	probability with which an agent changes from a worker to a non-worker	0.001
θ	initial response threshold (Eq. 1)	500
	response thresholds range (Eq. 1)	0–1000
ξ	coefficient for updating a response threshold when an agent is a non-worker (Eq. 2)	1
ψ	coefficient for updating a response threshold when an agent is a worker (Eq. 3)	10
Conventional method		
(δ, α)	combinations of (1,3), (3,5), (5,7), (7,9) and (9,11) were tested in simulation experiments (Eq. 4)	
Proposed method		
β	coefficient for scaling a contact stimulus with a foraging agent (Eq. 5)	1000
γ	attenuation rate for decreasing the contact stimulus over time (Eq. 6)	0.99

Table 3.
Parameters of simulation experiments.

4.2 Evaporation and diffusion of pheromones

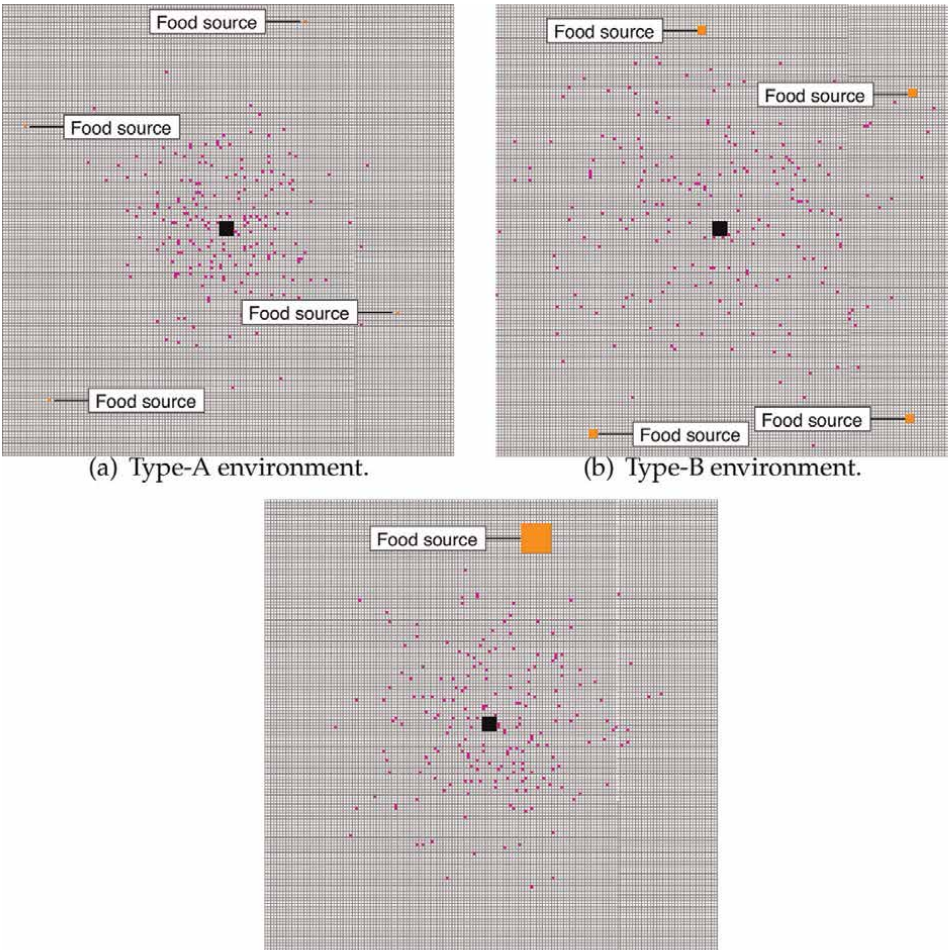
An agent secretes a trail or homing pheromone while moving. The initial value of each pheromone is 1.0, and each pheromone decreases at a rate of 0.99. The agent returns to the colony when the residual quantity of each pheromone is less than 0.01. These parameters were set through preliminary experiments so that an agent can sufficiently explore an environment. Here, an agent can explore an environment in approximately 450 steps.

Pheromones are spread out by evaporation and diffusion, diluting their density. The equations for these evaporation and diffusion phenomena are defined as follows.

$$F_p(x, y, t) = \mu F_p(x, y, t-1) + \Delta F_p(x, y, t) \tag{8}$$

$$\Delta F_p(x, y, t) = \begin{cases} Q_p & \text{if an agent is in the grid } (x,\ y) \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$\begin{aligned} a_p(x, y, t) = {} & a_p(x, y, t-1) + \lambda \Big\{ a_p(x+1, y-1, t-1) + a_p(x+1, y, t-1) \\ & + a_p(x+1, y+1, t-1) + a_p(x, y-1, t-1) + a_p(x, y+1, t-1) \\ & + a_p(x-1, y-1, t-1) + a_p(x-1, y, t-1) + a_p(x-1, y+1, t-1) \\ & - 9a_p(x, y, t-1) \Big\} + (1-\mu) F_p(x, y, t), \end{aligned} \tag{10}$$



(a) Type-A environment. (b) Type-B environment. (c) Type-C environment.

Figure 3.
Three types of environments, with varying amounts and distributions of feeds.

where Q_p denotes the addition quantity of the pheromone, and $F_p(x, y, t)$ represents the quantity of the pheromone in a grid (x, y) at a time, t. $a_p(x, y, t)$ represents the quantity of the pheromone above a grid (x, y) at a time, t. The second term on the right side of Eq. (10) represents the quantity of the pheromone that inflows, outflows and disappears from neighbouring grids, and the third term on the right side represents the quantity of the pheromone that evaporates. An agent detects the pheromone quantity, a_p, in the three forward cells. In the simulation, the initial pheromone density was set as $Q_p = 1.0$. γ and λ represent the rates of evaporation and diffusion, respectively, with values set as $\mu = 0.99$ and $\lambda = 0.01$. These parameters were set with reference to a previous study [19]. However, if the rates of evaporation and diffusion are very high, agents cannot arrive at a food source by following a pheromone trail because pheromones will disappear rapidly. Conversely, if the rates are very low, agents cannot discover pheromone trails leading to food sources because pheromones will fill the environment. Thus, we made appropriate adjustments to fit the simulation environment through preliminary experiments.

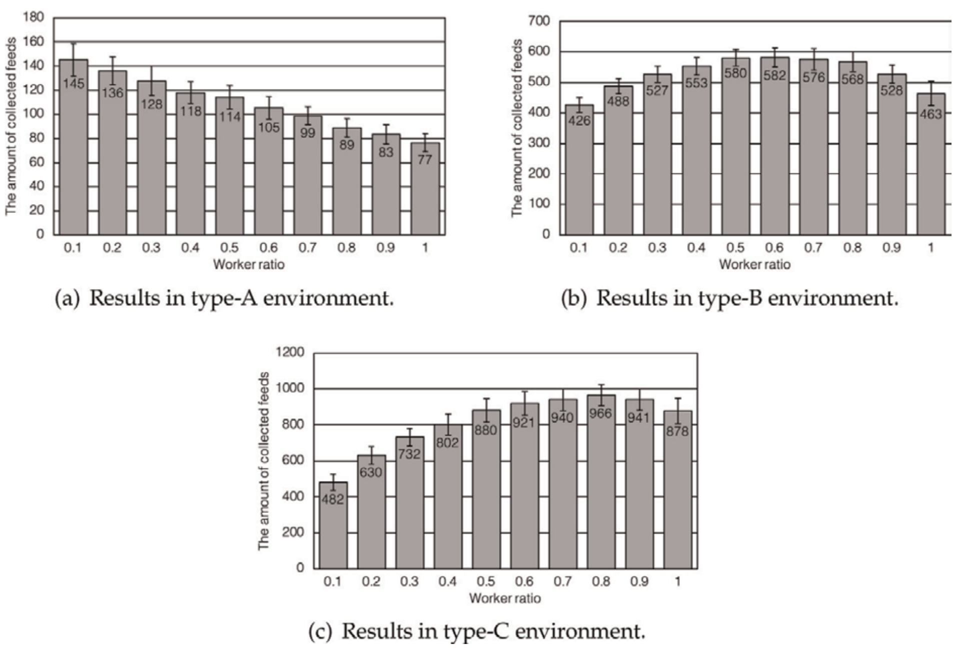
4.3 Simulation results

4.3.1 Appropriate worker ratio in each environment

Firstly, we reveal the appropriate worker ratio in the three types of environments with varying amounts and distributions of feeds. **Figure 4** depicts the means and standard deviations of collected feeds by a swarm of agents with different worker ratios in the three types of environments. The swarm of agents needed to increase the number of exploring agents because there was only a small amount of feed in the type-A environment, with four small food sources. Therefore, the mean of collected feeds was higher as the worker ratio was smaller. In the type-B environment, with four medium food sources, both foraging and food exploration were important for the swarm of agents. Therefore, the maximum mean of collected feeds was obtained when the worker ratio was 60%. In the type-C environment, with only one large food source, a swarm of agents could easily discover the large food source. Therefore, the swarm of agents needed to mobilise several ants to collect feeds efficiently. However, if all agents attended to the foraging call, it may take a long time to discover a new food source. Thus, the maximum mean of collected feeds was obtained when the worker ratio was 80%. According to the above results, the swarm of agents uses the appropriate worker ratio in each environment with varying amounts and distributions of feeds.

4.3.2 Adaptability of the proposed method

We studied the proposed method's adaptability in environments with varying amounts and distributions of feeds. In addition, we compared the proposed method



(a) Results in type-A environment.

(b) Results in type-B environment.

(c) Results in type-C environment.

Figure 4.
The means and standard deviations of collected feeds in different worker ratios.

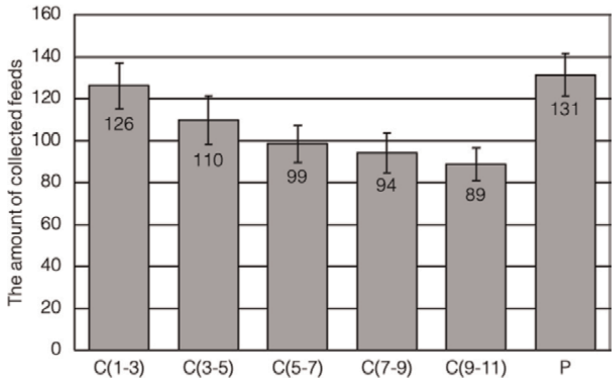
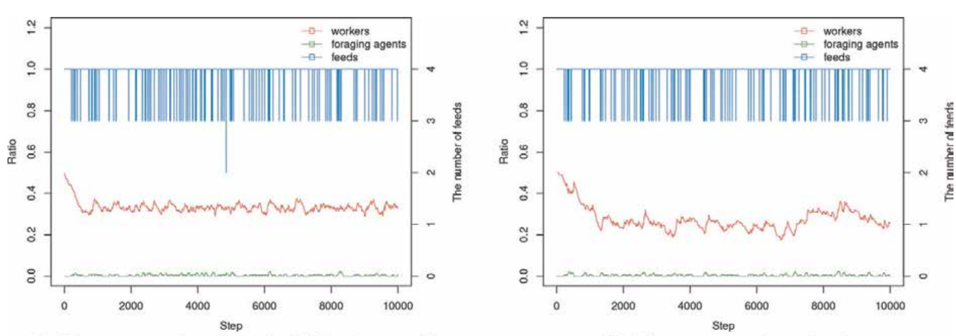


Figure 5.
Means and standard deviations of collected feeds in type-A environment.

with the conventional method in terms of the mean of collected feeds by a swarm of agents.

Figure 5 illustrates the means and standard deviations of collected feeds in the type-A environment. For example, C(1–3) denotes the conventional method, with the load parameter, δ and the scale factor, α, set as 1 and 3, respectively. On the other hand, P denotes the proposed method. Here, the maximum mean of collected feeds of the conventional method was 126. Similarly, the mean of collected feeds of the proposed method was 131. The conventional and proposed methods had the same foraging ability.

Figure 6 shows the relationship among the worker ratio, the foraging agent ratio and the amount of existing feeds in an environment. The horizontal axis represents the number of steps. The vertical axis represents the ratio of workers and foraging agents, and the secondary vertical axis represents the amount of existing feeds in the environment. In the simulation, when the feed in one source is exhausted, the next food source is placed randomly in the environment. Therefore, the amount of feeds fluctuated between 3 and 4. That is, the number of vertical blue lines represents the amount of collected feeds, and its slits represent the time spent discovering a new food source. **Figure 6** depicts the results of the proposed and conventional methods in terms of the amount of feeds collected. The worker ratio of the conventional method remained constant at approximately 30%, whereas that of the proposed method



(a) The conventional method (δ = 1, α = 3). (b) The proposed method.

Figure 6.
The transition of the worker ratio and the amount of existing feeds in type-A environment.

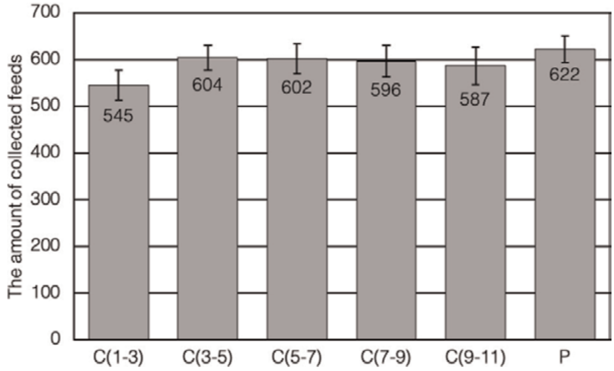


Figure 7.
Means and standard deviations of collected feeds in type-B environment.

fluctuated with the amount of existing feeds in the environment. In addition, the mean worker ratio of the proposed method was 28%. As a result, in both methods, the swarm of agents could discover a new food source quickly by increasing the number of exploring agents.

Figure 7 displays the means and standard deviations of collected feeds in the type-B environment. The maximum mean of collected feeds of the conventional method was 604. Here, the load parameter, δ and the scale factor, α, were set as 3 and 5, respectively. The mean of collected feeds of the proposed method was 622. As shown in **Figure 8**, the worker ratio of the conventional method remained constant at approximately 60%. On the other hand, the worker ratio of the proposed method increases when a new food source is placed in the environment, whereas its worker ratio decreases when the amount of existing feeds in the environment reduces. That is, with the proposed method, a swarm of agents could collect large amounts of feeds by adjusting the number of foraging and exploring agents according to the amount of existing feeds in the environment. Furthermore, the proposed method's mean worker ratio was 60%, which is mostly identical to that of the conventional method.

Figure 9 displays the means and standard deviations of collected feeds in the type-C environment. The maximum mean of collected feeds of the conventional method was 997. Here, the load parameter, δ and the scale factor, α, were set as 9 and 11, respectively. The proposed method's mean of collected feeds was 943. As shown in

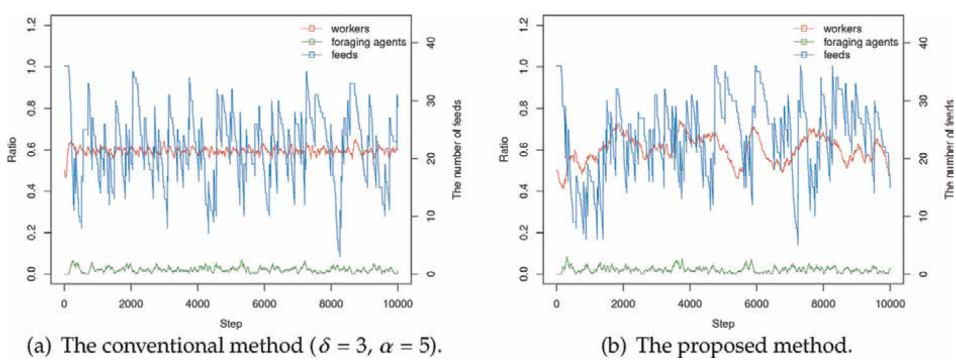


Figure 8.
The transition of the worker ratio and the amount of existing feeds in type-B environment.

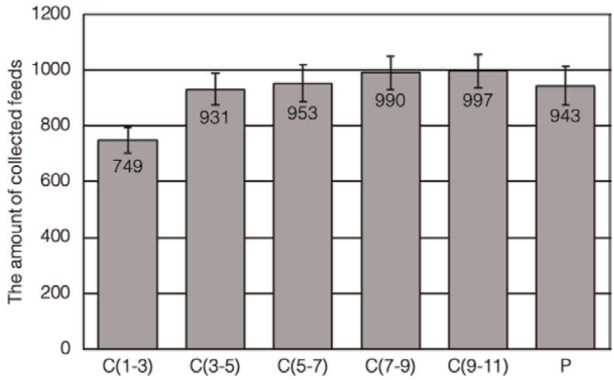


Figure 9.
Means and standard deviations of collected feeds in type-C environment.

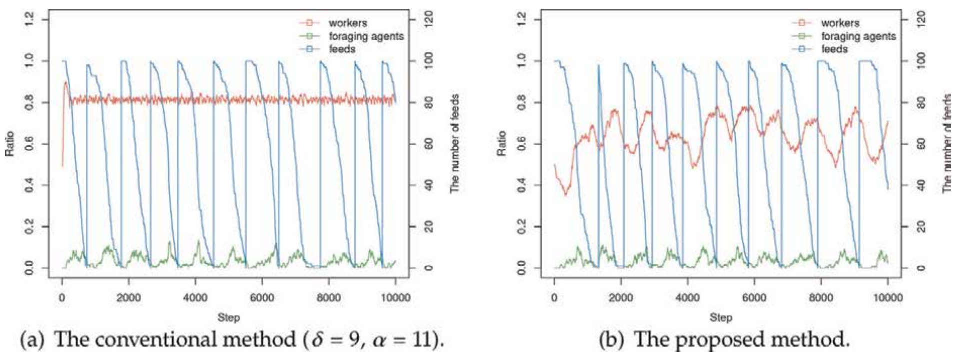
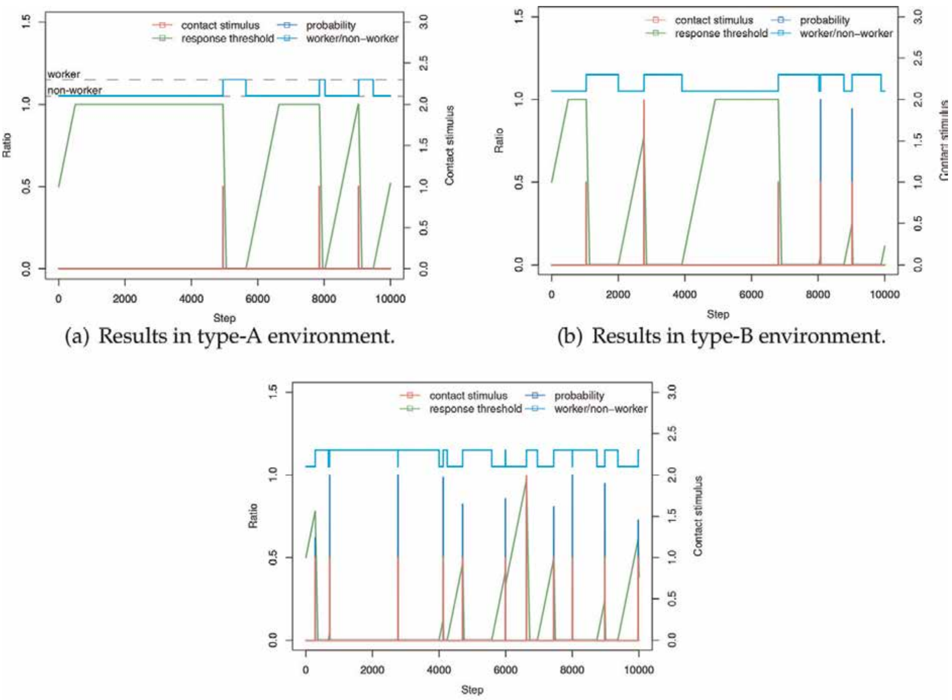


Figure 10.
The transition of the worker ratio and the amount of existing feeds in type-C environment.

Figure 10a, the conventional method's mean worker ratio is 80% because many workers must collect feed from one large food source effectively. On the other hand, the proposed method's mean worker ratio is 64%, as shown in **Figure 10b**. The reason for this was that, with the proposed method, the swarm of agents exhausted the feed in a food source before its worker ratio reached 80%.

4.3.3 Role assignment process

We explain the role assignment process in the proposed method in each environment with varying amounts and distributions of feeds. **Figure 11** depicts the relationship between the contact stimuli and the worker/non-worker state for a certain agent. In this graph, the horizontal axis represents the number of steps. The vertical axis represents the response threshold and the probability of changing from a non-worker to a worker. However, the value of the response threshold was normalised from 0.0 to 1.0. The secondary vertical axis represents the strength of the contact stimulus. The red line indicates the strength of contact stimuli with foraging agents. The green line indicates the transition of the response threshold. The blue line indicates the transition of the probability of changing from a non-worker to a worker. The light blue line indicates that an agent is in a worker or non-worker state. In addition, a convex shape indicates a worker state and a concave shape indicates a non-worker state. The



(a) Results in type-A environment.

(b) Results in type-B environment.

(c) Results in type-C environment.

Figure 11.
Relationship between contact stimuli and worker state in the proposed method.

probability of changing from a non-worker to a worker varied with the contact stimuli and the response threshold. On the other hand, the probability of changing from a worker to a non-worker remained constant at 0.1%.

As shown in **Figure 11a**, an agent remains in a non-worker state until approximately 5,000 steps, and the duration of the non-worker state is very long. The reason for this was that the frequency of contact stimuli with foraging agents was low because the type-A environment had only a small amount of feed. On the other hand, as shown in **Figure 11b**, an agent contacts foraging agents frequently because there is much feed in the environment, increasing the duration of the worker state. The agent contacted two foraging agents continuously at approximately 2,800 steps. Furthermore, the strength of the contact stimulus reached 1.99, and the agent changed to a worker. As shown in **Figure 11c**, an agent contacts foraging agents frequently after 4,000 steps and changes to a worker quickly even if it had changed to a non-worker. According to the aforesaid results, the proposed method can perform role assignment automatically under different environmental conditions through contact stimuli with foraging agents according to the amounts of feeds.

4.3.4 Robustness in a dynamic environment

To evaluate the proposed method's effectiveness in a dynamic environment, we compared the proposed method with the conventional method in terms of the mean of collected feeds in an environment with varying amounts and distributions of feeds. **Figure 12** depicts the means and standard deviations of collected feeds in a dynamic

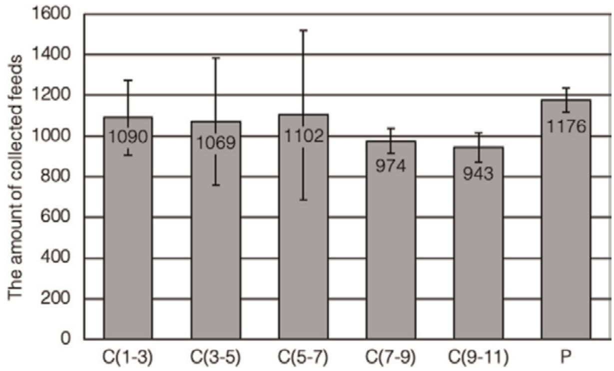


Figure 12.
Means and standard deviations of collected feeds in a dynamic environment.

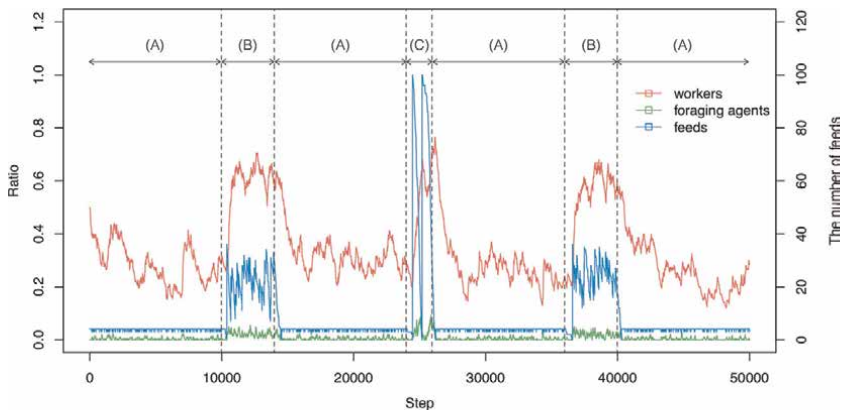


Figure 13.
The transition of worker ratio and the amount of collected feeds of the proposed method. (A), (B) and (C) in the graph denote an environment with four small food sources, an environment with four medium food sources and an environment with one large food source, respectively. The maximum number of steps in this simulation is 50,000.

environment. The conventional method's maximum mean of collected feeds was 1102, and its standard deviation was 414. Here, the load parameter, δ and the scale factor, α, were set as 5 and 7, respectively. Conversely, the proposed method's mean of collected feeds was 1176, and its standard deviation was 61. Based on the difference between standard deviations, the amount of collected feeds of the conventional method was unstable, whereas that of the proposed method was stable in the dynamic environment.

Figure 13 depicts the proposed method's results. The red line indicates the transition of the worker ratio, and the blue line indicates the amount of existing feeds in the environment. As shown in **Figure 13**, the worker ratio is low in the type-A environment with a small amount of feed. This means that the swarm of agents could discover new food sources by increasing the number of exploring agents. Conversely, the worker ratio was higher in the type-B and C environments, with more feeds. This means that the swarm of agents could collect much feed effectively by increasing the number of foraging agents. As a result, the proposed method could perform autonomous role assignment effectively through contact stimuli with foraging agents according to the external environment fluctuations. On the other hand, to change the worker/non-worker state, the conventional method used the stimulus updated by the previous stimulus, the load parameter and the worker ratio in the colony. This means that the strength of stimuli did not reflect the external environment fluctuations. Therefore, the conventional method did not allow agents to be assigned roles according to external environment fluctuations.

5. Conclusions

This study proposed a novel autonomous role assignment method using the response threshold model through local interactions in swarm robotic systems. In this study, to study the proposed method's robustness in a dynamic environment, we applied the proposed method to ant foraging problems, where the amount and distribution of feeds fluctuate in an environment. The conventional response threshold model, which mimics autonomous role assignment mechanisms of ants, uses the worker ratio in an ant colony as external stimuli. Conversely, the proposed method uses contact stimuli with foraging agents as external stimuli. Our simulations confirmed that, with the proposed method, agents can maintain an appropriate worker ratio and can effectively collect feeds in a dynamic environment compared with the conventional method. In addition, through the analysis of autonomous role assignment processes of ants, we revealed the mechanisms by which agents can specialise as a worker or a non-worker appropriately according to the external environment fluctuations through the frequency of contact stimuli with foraging agents. In future work, we will apply other mechanisms of cooperative behaviours of ants and bees, which are social insects, to swarm robotics systems. In addition, we will create a rescue robot team composed of many autonomous drones.

Author details

Kazuaki Yamada
Faculty of Science and Engineering, Department of Machine Engineering, Toyo University, Kujirai, Kawagoe-shi, Saitama, Japan

*Address all correspondence to: yamadak@toyo.jp

References

[1] Miller P. The Smart Swarm: How to Work Efficiently, Communicate Effectively, and Make Better Decisions Using the Secrets of Flocks, Schools, and Colonies. New York: Avery; 2000

[2] Navarro I, Matia F. An introduction to swarm robotics. ISRN Robotics. 2013; **2013**:608164

[3] Brambilla M, Ferrante E, Birattari M, Dorigo M. Swarm robotics: a review from the swarm engineering perspective. Swarm Intelligence. 2013;7:1-41. DOI: 10.1007/s11721-012-0075-2

[4] Dorigo M, Theraulaz G, Trianni V. Reflections on the future of swarm robotics. Science Robotics. 2020;**5**:49. DOI: 10.1126/scirobotics.abe4385

[5] Gordon DM. Ants at Work: How an Insect Society is Organized. New York: The Free Press; 1999

[6] Mitaka Y, Mori N, Matsuura K. Multi-functional roles of a soldier-specific volatile as a worker arrestant, primer pheromone and an antimicrobial agent in a termite. Proceedings of the Royal Society B: Biological Sciences. 2017; **284**(1859):20171134

[7] Bonabeau E, Theraulaz G, Deneubourg JL. Quantitative study of the fixed threshold model for the regulation of division of labour in insect societies. Biological Sciences. 1996; **263**(1376):1565-1569. DOI: 10.1098/rspb.1996.0229

[8] Bonabeau E, Sobkowski A, Theraulaz G, Deneubourg JL. Adaptive task allocation inspired by a model of division of labor in social insects. In: Proceedings of BCEC97. Singapore: World Scientific; 1997. pp. 36-45

[9] Theraulaz G, Bonabeau E, Denuebourg J-N. Response threshold reinforcements and division of labour in insect societies. Royal Society. 1998; **265**(1393):327-332. DOI: 10.1098/rspb.1998.0299

[10] Hasegawa E, Ishii Y, Tada K, Kobayashi K, Yoshimura J. Lazy workers are necessary for long-term sustainability in insect societies. Scientific Reports. 2016;**6**:20846

[11] Gordon DM, Mehdiabadi NJ. Encounter rate and task allocation in harvester ants. Behavioral Ecology and Sociobiology. 1999;**45**:370-377. DOI: 10.1007/s002650050573

[12] Greene MJ, Gordon DM. Interaction rate informs harvester ant task decisions. Behavioral Ecology. 2007;**18**(2):451-455. DOI: 10.1093/beheco/arl105

[13] Davidson JD, Arauco-Aliaga RP, Crow S, Gordon DM, Goldman MS. Effect of Interactions between harvester ants on forager decisions. Frontiers in Ecology and Evolution. 2016;**4**. Available from: https://pubmed.ncbi.nlm.nih.gov/28758093. DOI: 10.3389/fevo.2016.00115

[14] Yamada K. Specialization in swarm robotics using local interactions. International Journal of Engineering Research Technology (IJERT). 2016; **5**(10):103-108

[15] Yamada K. Autonomous role assignment and task allocation in scalable swarm robotic systems using local interactions. Artificial Life and Robotics. 2018;**23**(4):636-644. DOI: 10.1007/s10015-018-0500-8

[16] Nakayama K, Sueoka Y, Ishikawa M, Sugimoto Y, Osuka K. Control of

transportation trails by distributed autonomous agents inspired by the foraging behavior of ants. Nonlinear Theory and Its Applications. 2014;**5**(4): 487-498. DOI: 10.1587/nolta.5.487

[17] Ogihara Y, Yamanaka O, Akino T, Izumi S, Awazu A, Nishimori H. Switching of Primarily Relied Information by Ants: A Combinatorial Study of Experiment and Modeling, Mathematical Approaches to Biological Systems: Networks, Oscillations and Collective Motions. New York: Springer; 2015. pp. 119-137. DOI: 10.1007/978-4-431-55444-8_6

[18] Iwai T, Wakamiya N, Murata M. Response threshold model-based device assignment for cooperative resource sharing in a WSAN. International Journal of Swarm Intelligence and Evolutionary Computation. 2012;**1**: 235560

[19] Fujisawa R, Imamura H, Hashimoto T, Matsuno F. Communication using pheromone field for multiple robots. In: Proceedings of IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems (IROS2008). New York: IEEE; 2008. pp. 1391-1396. DOI: 10.1109/IROS.2008.4650971

Edited by Marco Antonio Aceves-Fernández

Artificial intelligence (AI) has attracted the attention of many and is taking an increasingly crucial role in our modern society. AI machines demonstrate advanced cognitive skills in making decisions, learning, perceiving the environment, predicting certain behavior, and processing written or spoken languages, among other skills. In tandem with AI, "swarm intelligence" is being used to solve complex problems that are beyond the reasoning and problem-solving abilities of humans. This book discusses both AI and swarm intelligence, including their theoretical backgrounds and applications.

Andries Engelbrecht, Artificial Intelligence Series Editor

Published in London, UK

IntechOpen

ISSN 2633-1403
ISBN 978-1-83768-088-7
9 781837 680887