Research article

Collaborative beamforming in wireless sensor networks using a novel particle swarm optimization algorithm variant

Robert Macharia Maina^{a,b,*}, Philip Kibet Lang'at^{a,b}, Peter Kamita Kihato^{a,c}

^a Institute for Basic Sciences Technology and Innovation, Pan African University, Nairobi, Kenya

^b Department of Telecommunication and Information Engineering, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya

^c Department of Electrical and Electronic Engineering, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya

ARTICLE INFO

Keywords:

Wireless sensor network
Collaborative beamforming
Particle swarm optimization
Beamsteering

ABSTRACT

Collaborative Beamforming (CBF) is an essential tool towards increasing transmission range in Wireless Sensor Networks (WSNs). Owing to the random and complex nature of WSNs, development and use of improved metaheuristic algorithms in CBF is of essence. Particle Swarm Optimization (PSO) algorithm is a good candidate for use in CBF owing to its simplicity and low computation complexity. However, the basic PSO algorithm suffers from premature convergence particularly in highly multimodal functions (typical of CBF). This paper delves into the development and application of an improved Particle Swarm Optimization (PSO) algorithm in CBF. A new fuzzy-logic based confidence and inertia weight parameters adaptation scheme has been developed with an aim of enhancing exploration and exploitation capabilities of the PSO algorithm. Normalized particle quality and iteration count have been used as the inputs to the designed fuzzy-logic inference system. The fuzzy logic based parameters adaptation scheme has been implemented in the form of a lookup table to minimize “on-line” computation complexity. Furthermore, a particle culling/ re-initialization procedure is utilized at half the number of maximum iterations to enhance overall swarm diversity. The modified PSO algorithm has been christened Cuffed Fuzzy Adaptive Particle Swarm Optimization (CFAPSO) algorithm. The developed CFAPSO algorithm is noted to outperform other metaheuristic algorithms in a statistical performance analysis procedure (on the basis of a set of standard unimodal and multimodal functions). Upon application to CBF, the CFAPSO algorithm is found to generate a beamsteering outcome statistically identical to that of conventional beamsteering.

1. Introduction

A Wireless Sensor Network (WSN) is an interconnection of wirelessly linked, small-sized and low power sensor nodes spread out (usually randomly) over an area of monitoring interest. Unlike traditional sensors, WSN nodes are equipped with data processing and wireless transmission capabilities. Reference can be made to [1, 2, 3, 4, 5, 6].

Sensor nodes transmit sensed data to remote sinks (for final processing/ analysis or for onward forwarding to other specialized processing/ analysis/ storage terminals). Fig. 1 illustrates the WSN concept.

Sensor nodes in WSNs bear limited energy sources implying limited data transmission range (*required transmission power increases with transmission distance as per the inverse square law*). Long-range transmission consequently necessitates multiple hop transmission resulting in increased network interference. A recent solution to the long-range transmission problem is Collaborative Beamforming (CBF).

A typical collaborative beamforming scenario is as depicted in Fig. 2. The desire is to form a link between a source node (one of the collaborating nodes) and the sink. In this case, it is assumed that the sink is out of the source node's transmission range. The source node shares the data it ought to transmit with carefully selected nodes in its neighbourhood (collaborating nodes). The collaborating nodes coherently form a transmission link towards the sink through appropriate amplitude and phase weighting at individual nodes (CBF). The outcome is a radiation pattern whose main beam is ideally oriented towards the sink. Reference can be made to [7].

CBF with a set of N collaborating nodes each transmitting at a fixed power can extend the transmission range by N . The transmission power can be reduced to the extent of $\frac{1}{N^2}$ per node given a fixed transmission range [7].

CBF schemes are widely classified into two categories: feedback based and non-feedback based. Feedback based schemes tend to in-

* Corresponding author at: Institute for Basic Sciences Technology and Innovation, Pan African University, Nairobi, Kenya.

E-mail address: robertisaacm@gmail.com (R.M. Maina).

<https://doi.org/10.1016/j.heliyon.2021.e08247>

Received 14 August 2021; Received in revised form 2 October 2021; Accepted 21 October 2021

Fig. 1. A typical WSN: Sinks (circular shape), sensor nodes (star shape).

Fig. 2. A typical collaborative beamforming scenario. Collaborating nodes are highlighted in gray. The main beam is oriented towards the sink.

crease network load, hence the wide adoption of non-feedback based schemes.

The delay-and-sum beamforming concept typical of conventional antenna array beam-steering has traditionally been applied in CBF [7]. Delay-and-sum beamforming cannot be directly utilized in beam pattern optimization in CBF despite being a plausible candidate for beamsteering. Beam pattern optimization calls for application of deterministic or stochastic algorithms in beamforming weights selection.

1.1. Related work: application of Artificial Intelligence (AI) algorithms in CBF

There is growing interest in research in application of Artificial Intelligence (AI) algorithms in CBF. In [8], research into the use of Cuckoo Search (CS) algorithm in reducing the maximum sidelobe level in a CBF process is highlighted. A discrete version of the CS algorithm is utilized in node location selection and a continuous version of the CS algorithm is utilized in optimization of the excitation amplitude in the selected nodes. The proposed CS algorithm based approach is noted to outperform cross-entropy optimization based approach in sidelobe level minimization. In [9], use of Variation Particle Chicken Swarm Optimization (VPCSO) algorithm in maximum sidelobe suppression in CBF is presented. The VPCSO algorithm is utilized in optimization of the CBF nodes' amplitude. The algorithm is noted to outperform Biogeography Based Optimization (BBO) and Chicken Swarm Optimization (CSO) algorithms. In [10] a Distributed Parallel Cuckoo Search (DPCS) algorithm is utilized in CBF optimization. The optimization problem entails minimizing the maximum sidelobe level, node moving energy consumption and transmission power in a mobile WSN. The DPCS is noted to perform better than standard CS algorithm, Firefly algorithm (FA) and Genetic algorithm (GA). In [11], use of an Improved Non-dominated Sorting GA algorithm (INSGA) and a distributed parallel INSGA in CBF optimization is presented. In [12], a node selection based algorithm for CBF in WSNs is presented. This is achieved through application of a low complexity greedy node selection algorithm. The work gener-

ally highlights the necessity of low complexity algorithms in CBF in WSNs. However, the adopted node selection procedure is only feasible in highly dense WSNs. The adopted greedy node selection algorithm is sub-optimal, and is bound to take an extended duration to yield feasible CBF results. The work does not address CBF in 3-dimension node/sink layout. In [13], a review of research in CBF in WSNs is presented. Research trends covered are in the domains of beam-pattern oriented CBF mechanisms, node power and node lifetime optimization, node transmission phase synchronization and practical CBF implementation. Notably, among the recommendations made for future research is CBF in the case of 3-dimension node/sink layout.

Current research in CBF is generally in the domain of planar node arrangement with the sink at the nodes' plane. It is worth researching into CBF given planar node arrangement with the sink at an elevated plane; This is taken into consideration in this paper.

1.2. Related work: research work in PSO algorithm

Among metaheuristic optimization algorithms in place, Particle Swarm Optimization (PSO) algorithm stands out owing to its simplicity, fast convergence and outstanding overall performance. PSO algorithm has been applied in a plethora of problems since its inception.

Despite its good performance in handling a number of optimization problems, the basic PSO has some shortcomings: premature convergence (getting trapped in local optima) particularly in large scale multimodal complex optimization problems. This can be attributed to poorly tuned parameters (inertia/ confidence) and the basic global updating strategy that typically yields loss of diversity in the swarm. Consequently, research has been dedicated towards averting the premature convergence issue. Recent research entailing modification and application of the PSO algorithm in various problems can be found in [14, 15, 16, 17, 18, 19, 20, 21, 22].

The proposed PSO algorithm variants are generally geared towards balancing local and global search capabilities. This is mainly through inertia weight/ confidence parameter tuning, modification of the velocity/ position update rules, swarm topology perturbation, application of evolution strategies (for example, culling/ mutation) among others. Nevertheless, there is still room for new (problem-specific) proposals with recent application areas (optimization problems) becoming increasingly complex.

In [14], a mutation operator is introduced with the aim of overcoming the "local trap" issue. The mutation operator inherently improves the swarm particles' diversity. In general, an improvement in convergence speed and performance stability is observed. A hybrid PSO algorithm employing an Adaptive Learning Strategy (ALPSO) is designed in [16]. A self-learning based candidate solution generation strategy aimed at improving exploration and a competitive learning based strategy aimed at improving exploitation are brought to the fore. Moreover, a tolerance based search direction adjustment scheme aimed at balancing exploration and exploitation is developed. An experimental analysis based on a set of 40 benchmark test functions establishes the strength of the proposed improvements on performance accuracy and convergence speed. In [17], a Hybrid Many-objective PSO (HMaPSO) algorithm is designed with the platform of application being a multi-objective green coal production problem. Offspring generation and selection mechanisms typical of evolutionary strategies are applied yielding appreciable performance improvement. A hybrid Firefly-PSO (HFPPO) algorithm that takes advantage of the strengths of basic Firefly and PSO algorithms is proposed in [18]. In the proposed algorithm, the starting point of local search is determined intelligently by checking into previous global best fitness values. A statistical performance comparison brings out the strengths of the proposed algorithm in handling unimodal, multimodal alongside other computationally expensive numerical functions. In [20], a Differential Evolution-Crossover Quantum PSO (DE-CQPSO) algorithm is designed. The developments are on the basis of the fast convergence associated with DE algorithm and the particle diversity typical

Fig. 3. Wireless sensor network model illustration (a 3-dimension perspective from the point of view of the sinks' location relative to the nodes' plane). Sensor nodes: *star shape*; Sinks: *circular shape*; cluster head: *rectangular shape*.

of crossover operators in genetic algorithm. In general, an improvement in convergence speed is noted.

An adaptive parameter approach entailing sampling parameter values from a promising set is developed in [23]. The scheme eliminates the necessity of specifying and tuning PSO parameters. A set of three adaptive strategies are proposed in [24]. Among the proposed strategies entails parameter tuning. A comparative analysis brings to the fore the strengths of the proposed scheme. In [25] an adaptive approach utilizing Lévy flights to enhance swarm diversity/ reduce stagnation at local optima is presented. The proposed approach outperforms the standard PSO algorithm.

The afore-cited research brings to the fore the significance of use of improved variants of AI algorithms in correspondence to the optimization problem under consideration. There is room for research in problem-specific improvements with the ever increasing problem complexity. An appropriately improved PSO algorithm befits application in CBF owing to its simplicity and low computation complexity. In this paper, fuzzy logic has been applied in PSO inertia weight and confidence parameters adaptation (upon carrying out a problem specific meta-optimization procedure). An off-line approach towards implementing the fuzzy-adaptive parameter values is proposed to minimize overall computation complexity. This is alongside application of selection and culling mechanisms typical of evolution strategies (in particular genetic algorithm). The modified PSO algorithm (Culled Fuzzy Adaptive PSO (CFAPSO) algorithm) is noted to outperform other algorithms in solving the CBF problem.

The rest of this paper is organized as follows. A description of collaborative beamforming is given in Section 2. The standard PSO algorithm, genetic algorithm and fuzzy logic concepts are described in Section 3. The proposed CFAPSO algorithm is presented in Section 4. An analysis of the performance of the CFAPSO algorithm on several benchmark unimodal and multimodal functions is given in Section 5. The CFAPSO algorithm is applied in CBF in Section 6. The overall conclusion is given in Section 7.

2. Collaborative beamforming array factor

The CBF array factor corresponding to a planar node layout with the sink at an elevated plane is herein presented.

A WSN model featuring 2-dimension random arrangement of sensor nodes (as per the layout in Fig. 3) is utilized. This modelling approach is sufficient in representing scenarios in which sensor nodes are deployed on planar/ nearly-planar surfaces.

The model takes into consideration a large variety of practical WSNs deployment scenarios. Practically, the 3-dimension perspective in terms of sink placement might correspond to an Unmanned Aerial Vehicle (UAV) based sink [26].

With reference to Fig. 3, through basic trigonometry, the distance between the k^{th} node and the sink (A_0, ϕ_0, θ_0) is as per Eq. (1).

$$d_k |_{(A_0, \phi_0, \theta_0)} = \sqrt{A_0^2 + r_k^2 - 2r_k A_0 \sin \theta_0 \cos(\phi_0 - \psi_k)} \quad (1)$$

In general, with reference to some general position (A, ϕ, θ) , the relationship in Eq. (2) holds true.

$$d_k |_{(A, \phi, \theta)} = \sqrt{A^2 + r_k^2 - 2r_k A \sin \theta \cos(\phi - \psi_k)} \quad (2)$$

In the far-field radiation region, the relationship $A \gg r_k$ holds. Consequently, d_k can be approximated as per Eq. (3).

$$d_k |_{(A, \phi, \theta)} \approx A - r_k \sin \theta \cos(\phi - \psi_k) \quad (3)$$

The array factor for a total of K collaborating nodes, each transmitting at an amplitude equivalent to $1/K$ is as per Eq. (4).

$$AF_{\phi, \theta} = \frac{1}{K} \sum_{k=1}^K e^{j\psi_k} e^{j\frac{2\pi}{\lambda} d_k(\phi, \theta)} \approx \frac{1}{K} \sum_{k=1}^K e^{j\psi_k} e^{j\frac{2\pi}{\lambda} [A - r_k \sin \theta \cos(\phi - \psi_k)]} \quad (4)$$

Where:

- AF : Array Factor.
- K : Total number of collaborating nodes.
- Ψ_k is the initial phase for node $k \in 1, 2, 3 \dots K$.
- Node transmission amplitude: $1/K$.

Eq. (4) can be re-written in the form of Eq. (5).

$$AF_{\phi, \theta} \approx \frac{1}{K} \sum_{k=1}^K e^{j\psi_k} e^{j\frac{2\pi}{\lambda} A} e^{j\frac{2\pi}{\lambda} [-r_k \sin \theta \cos(\phi - \psi_k)]} \quad (5)$$

The term $e^{j\frac{2\pi}{\lambda} A}$ is a phase offset.

The node transmission amplitude ($1/K$) and phase (Ψ_k) can be varied to change the form of the array factor; for instance to steer the main beam of the array factor towards a desired direction.

The terms $1/K$, $e^{j\psi_k}$ and $e^{j\frac{2\pi}{\lambda} A}$ can be combined to form a single variable complex weight (w_k) as per Eq. (6).

$$AF_{\phi, \theta} \approx \sum_{k=1}^K w_k e^{j\frac{2\pi}{\lambda} [-r_k \sin \theta \cos(\phi - \psi_k)]} \quad (6)$$

Upon normalizing r_k with respect to signal wavelength ($\bar{r}_k = r_k / \lambda$), Eq. (6) can be expressed as per Eq. (7). The wavelength normalization procedure so undertaken allows for wavelength independent CBF problem formulation.

$$AF_{\phi, \theta} \approx \sum_{k=1}^K w_k e^{j2\pi [-\bar{r}_k \sin \theta \cos(\phi - \psi_k)]} \quad (7)$$

Generally, the array factor as presented in Eq. (7) can be manipulated by optimally adjusting the node transmission complex weights (w_k) to fit a desired form/ pattern (*the CBF action*).

3. Algorithm reviews

3.1. Particle swarm optimization algorithm

PSO algorithm is essentially a checked movement of a swarm of “particles” (*representative of potential solutions*) in some defined search space. Eqs. (8) and (9) are representative of the PSO algorithm.

$$v_i(t+1) = (wv_i(t)) + (c_p r_p (p_i - x_i)) + (c_s r_s (l_i - x_i)) \quad (8)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (9)$$

For some particle i at iteration t and position x_i , the particle's velocity v_i is updated in accordance to Eq. (8). Inertia weight w controls the influence of the immediate previous velocity. It is commonly decreased linearly from 0.9 to 0.4 [27]. Parameter c_p (self/ personal confidence parameter) controls the influence of the personal best position (p_i) achieved. It is commonly set at a static value of 2 [28]. Parameter c_s

(social confidence parameter) controls the influence of the best position that has been found by any of the particles in the neighbourhood of particle i (l_i). It is commonly set at a static value of 2 [28]. Random values r_p and r_s are usually drawn with uniform probability from the number set $[0 - 1]$. The random values serve to increase the exploration capabilities of personal and social influences. The particle position x_i is updated according to Eq. (9). Reference can be made to [29, 30, 31].

3.2. Standard genetic algorithm

Genetic Algorithm (GA) is an evolutionary metaheuristic inspired by the natural process of evolution/ selection (*survival for the fittest*) [32]. The action of GA can be summarized in the following main steps [32]:

1. The starting point is the creation of a random initial population. Each member of the population is a *candidate solution*, typically represented in gray coded binary strings.
2. Each member of the population (*candidate solution*) is assigned a score corresponding to its fitness value as per the cost function under consideration (*raw fitness scores*).
3. The *raw fitness scores* are subsequently converted (scaled) to an appropriate range of values, *expectation values*.
4. Some population members are selected stochastically on the basis of *expectation values* to form a *parent population*. *Elite individuals*, the individuals that are passed to the next population, are also selected on the basis of *expectation values*.
5. *Children* are subsequently generated from the *parent population*. The *children* generation process is through either *mutation* (making random changes to a single *parent*) or *crossover* (combining the vector entries of a pair of *parents*).
6. The current population is consequently replaced with the *elite individuals* and *children* to form the next generation.
7. Steps 2 to 6 are repeated over again a number of times in an attempt at getting better and better solutions. The algorithm is terminated once the termination condition is met.

Population size, *mutation* and *crossover* probabilities are usually appropriately tuned to enhance the solution generation process. Too high a mutation/ crossover rate is bound to lead to loss of good solutions. Too low a mutation rate is bound to lead to low population variation.

3.3. Fuzzy logic

Fuzzy logic involves approximate rather than exact reasoning/ computation. Fuzzy logic maps an input data space to some output space via approximate mathematical functions. Fuzzy logic can easily generate viable non-linear mapping functions in a variety of logic/ mathematical problems [33, 34]. Fuzzy logic exploits tolerance for uncertainty, approximation, imprecision and partial truth to achieve tractability, robustness and low solution cost.

4. Modification of particle swarm optimization algorithm

4.1. PSO parameter selection

The choice of PSO confidence parameter values greatly affects the overall algorithm performance. Personal and social confidence parameter values ought to be selected judiciously in line with the problem under consideration. In the case at hand, controlled brute-force search is applied in PSO self and social confidence parameter values selection.

A beamsteering problem is utilized in the meta-optimization process. A swarm size of 30 particles is considered, with 60 algorithm iterations. Brute-force search is run over social and self confidence values within the interval $[0 \text{ to } 4]$ in steps of 0.1. The obtained outcome is as per Fig. 4. The best confidence parameter combination is identified as a self/ personal-confidence parameter (C_1) value of 2.4 and a social-confidence parameter (C_2) value of 2.2.

Fig. 4. Meta-optimization surface in the form of a contour plot. The ideal set of confidence parameter values is marked with a cross symbol (2.4 against 2.2).

The standard PSO algorithm is utilized in subsequent sections with the following parameters: a swarm size of 30 particles, an inertia weight value ranging from 0.9 to 0.4 (decreased linearly during a run) [27], a self/ personal-confidence parameter (C_1) value of 2.4 and a social-confidence parameter (C_2) value of 2.2.

4.2. Fuzzy logic based PSO parameters adaptation scheme

4.2.1. Adaptive PSO algorithm parameters

The general idea laid out in this section pertains striking a balance between search space exploration and exploitation. Exploration is essentially delineated by the extent to which swarm particles search the entire defined search space. Exploitation is essentially delineated by the extent to which swarm particles search a defined high potential region. PSO algorithm parameter values directly affect exploitation and exploration capabilities of a swarm.

4.2.1.1. Adaptive inertia weight A large inertia weight value is ideal in exploration stages and a smaller value in exploitation stages.

In previous work, time and iteration varying inertia weight in the range $[0.4 - 0.9]$ has been recommended. Time and iteration based procedures do not take into account the solution quality corresponding to each and every swarm particle. It would be ideal to tie inertia weight value to particle quality.

The adaptive inertia weight procedure presented herein ties inertia weight value to particle quality as per the associated cost function value; this is alongside iteration dependence. Normalized particle quality (Ψ_i) is obtained as per Eq. (10). This evaluation is to be performed at each and every algorithm iteration.

$$\Psi_i = \frac{CF_{\max} - CF_i}{CF_{\max} - CF_{\min}} \quad (10)$$

In Eq. (10): CF denotes cost function value, CF_i denotes cost function value corresponding to particle i , $CF_{\max}$ denotes maximum cost function value attained in the iteration under consideration and $CF_{\min}$ denotes minimum cost function value attained in the iteration under consideration.

A high value of Ψ_i is indicative of a particle bearing a solution near the global best (taking into consideration a minimization problem) and vice-versa. Particles associated with low valued Ψ_i are accorded high inertia weight values to enhance their exploration. Particles associated with high valued Ψ_i are accorded low inertia weight values to enhance their exploitation. Iteration-wise, inertia weight is reduced with increase in algorithm iteration count. The mapping between parameter Ψ_i , iteration count and inertia weight (in the range $[0.4 - 0.9]$) is done on the basis of a Fuzzy Logic system (Section 4.2.2).

Fig. 5. Proposed fuzzy logic based PSO parameters adaptation scheme.

4.2.1.2. Adaptive personal and social confidence parameters Large values of c_p as compared to c_s result in enhanced exploration. Large values of c_s as compared to c_p result in enhanced exploitation and convergence. Consequently, it is imperative to have adaptive confidence parameters as opposed to the static values defined in the standard PSO algorithm.

Herein, the value of the social confidence parameter (c_s) is mapped onto the range [2 - 2.4] on the basis of parameter Ψ_i as defined in Eq. (10) and iteration count. The value of the self/ personal confidence parameter (c_p) is mapped onto the range [2.2 - 2.6] on the basis of parameter Ψ_i as defined in Eq. (10) and iteration count. These intervals are informed to a great extent by the meta-optimized PSO algorithm presented in Section 4.1. The mapping between parameter Ψ_i , iteration count and confidence parameters (c_p and c_s) is done on the basis of a Fuzzy Logic system (Section 4.2.2).

4.2.2. Fuzzy system design

The proposed mapping between parameter Ψ_i , iteration count and inertia weight/ confidence parameters (c_p and c_s) as per Section 4.2.1 is done on the basis of a fuzzy logic inference system, the subject of this section. The developed fuzzy logic based PSO parameters adaptation scheme is as per the illustration in Fig. 5.

The fuzzy logic based PSO parameters adaptation scheme has been implemented in the form of a look-up table to minimize “on-line” computation complexity. A Mamdani FIS is utilized, with other setup features being: *minimum* AND method, *minimum* Implication method, *maximum* Aggregation method and *centroid* Defuzzification method.

4.2.2.1. Variable fuzzification The input variables (normalized particle performance index/ quality (Ψ_i) and iteration count values) are interpreted as either *Small*, *Medium* or *Large*. The output variables (inertia weight, personal and social confidence values) are assigned the linguistic terms *Very Small*, *Small*, *Medium*, *Large* and *Very Large*. Gaussian membership functions have been utilized owing to their “accurate” representation of the input-output relationships implying increased evaluation reliability. The normalized particle performance index (Ψ_i) membership functions are as per Fig. 6. The normalized iteration count membership functions follow an arrangement equivalent to that presented in Fig. 6. The inertia weight membership functions are as per Fig. 7. The inertia weight values fall within the limits [0.4 → 0.9]. The personal confidence membership functions are a replica of Fig. 7 but within the limits [2.2 → 2.6]. Similarly, the social confidence membership functions are as per Fig. 7 but within the limits [2 → 2.4].

4.2.2.2. Fuzzy rules design The designed fuzzy rules corresponding to the inertia weight and social confidence variables are as presented in Tables 1 and 2. Fuzzy rules corresponding to the personal confidence variable are a replica of Table 1.

4.2.2.3. Fuzzy inference system results The relationship obtained between the fuzzy inference system input variables (quality parameter Ψ_i , iteration count) and the inertia weight output variable is as depicted in Fig. 8.

The relationship obtained between the fuzzy inference system input variables (quality parameter Ψ_i , iteration count) and the personal confidence output variable is a replica of Fig. 8 but on a scale of [2.2 → 2.6].

Fig. 6. Quality parameter (Ψ_i) membership functions.

Fig. 7. Inertia weight membership functions.

Table 1. Fuzzy rules corresponding to the inertia weight output variable.

		Number of iterations		
		Small	Medium	Large
Parameter Ψ_i	Small	Very Large	Large	Medium
	Medium	Large	Medium	Small
	Large	Medium	Small	Very Small

Table 2. Fuzzy rules corresponding to the social confidence output variable.

		Number of iterations		
		Small	Medium	Large
Parameter Ψ_i	Small	Very Small	Small	Medium
	Medium	Small	Medium	Large
	Large	Medium	Large	Very Large

Fig. 8. A contour plot indicating the relative relationship between quality parameter Ψ_i , iteration count and the resultant inertia weight value.

confidence output variable is a replica of Fig. 8 but on a scale of [2.2 → 2.6].

The relationship obtained between the fuzzy inference system input variables (quality parameter Ψ_i , iteration count) and the social confidence output variable is as depicted in Fig. 9.

Table 3. Selected inertia weight values (rounded up to 2 decimal places).

Normalized iterations	1	0.65	0.64	0.62	0.60	0.58	0.57	0.55	0.54	0.51	0.49	0.47
	0.9	0.66	0.65	0.63	0.61	0.60	0.58	0.56	0.55	0.53	0.51	0.49
	0.8	0.68	0.67	0.65	0.63	0.61	0.60	0.58	0.56	0.55	0.53	0.51
	0.7	0.70	0.69	0.67	0.65	0.63	0.62	0.60	0.58	0.56	0.55	0.54
	0.6	0.72	0.70	0.69	0.67	0.65	0.64	0.62	0.60	0.58	0.56	0.55
	0.5	0.73	0.72	0.70	0.68	0.66	0.65	0.64	0.62	0.60	0.58	0.57
	0.4	0.75	0.74	0.72	0.70	0.68	0.66	0.65	0.63	0.61	0.60	0.58
	0.3	0.76	0.75	0.74	0.72	0.70	0.68	0.67	0.65	0.63	0.61	0.60
	0.2	0.79	0.77	0.75	0.74	0.72	0.70	0.69	0.67	0.65	0.63	0.62
	0.1	0.81	0.79	0.77	0.75	0.74	0.72	0.70	0.69	0.67	0.65	0.64
	0	0.83	0.81	0.79	0.76	0.75	0.73	0.72	0.70	0.68	0.66	0.65
		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
		Solution quality

Fig. 9. A contour plot indicating the relative relationship between quality parameter Ψ , iteration count and the resultant social confidence value.

It is of essence to maintain computation complexity at a low during inertia weight, personal and social confidence parameter values adaptation. Consequently, a subset of the fuzzy inference generated inertia weight, personal and social confidence parameter values are selected for use in a “look-up table” function. For instance, selected inertia weight values are as per Table 3.

Subsets of the fuzzy inference generated personal and social confidence values are selected for use in a “look-up table” function in a manner similar to that presented in Table 3.

4.2.2.4. Look-up table operation At each and every iteration, PSO algorithm inertia weight, personal and social confidence values ought to be adapted in accordance to solution quality and normalized iterations as pinpointed earlier in Section 4.2.1. The utilized parameter adaptation procedure is as listed below.

1. Rounding up the normalized iterations value to 1 decimal place.
2. Rounding up the solution quality (performance index) value to 1 decimal place.
3. Performing look-up table operations (e.g. on Table 3) to obtain the values corresponding inertia weight, personal and social confidence.¹

4.3. Design of a low complexity hybridization scheme with an evolution based algorithm

The utilized hybridization scheme borrows aspects of culling from the GA algorithm. The culling scheme (to be invoked at half the number of maximum algorithm iterations) is as per the listing below.

¹ Algorithm wise, the look-up table operation is on the basis of extracting a matrix value corresponding to the row and column associated with the rounded up normalized iterations and solution quality values.

1. The swarm/ population particles (*candidate solutions*) are ranked in accordance to fitness scores as per the cost function under consideration.
2. Particles corresponding to the bottom half in terms of fitness scores are culled.
3. Particles corresponding to the top half in terms of fitness scores (*elite individuals*) are passed to the next population.
4. New particles are randomly created to refill the next population (substitute culled particles). This mimics the aspect of (*children*) generation process in the GA algorithm.

4.4. Adoption of a viable “local best” neighbourhood scheme

Use of a number of local neighbourhoods rather than a single global neighbourhood leads to enhanced swarm exploration. The adopted neighbourhood scheme is static in nature wherein the entire swarm is divided into 4 local neighbourhoods. This simplistic approach is aimed at ensuring minimal computation complexity.

4.5. PSO algorithm implementation in the special case of functions of complex variables

The general PSO algorithm structure has been altered to allow for optimization of the envisaged complex valued functions. A complex variable can be presented as per Eq. (11) (this matches the complex weights desired in the proposed CBF schemes). With Eq. (11) in mind, the approach taken is that of having each and every swarm particle framed in dimensions A and B . In the proposed CBF schemes, dimensions A and B are vectors of identical length, the length being equivalent to the number of nodes participating in CBF.

$$W = A \exp^{jB} \quad (11)$$

4.6. Modifications summary

The overall structure of the modified PSO algorithm, christened CFAPSO algorithm, is as per the flowchart in Fig. 10.

The general essence of the modifications carried out is captured in Fig. 11.

5. Performance analysis of the CFAPSO algorithm

Herein, the performance of the developed CFAPSO algorithm is analyzed against that of APSO, Basic PSO, GA, Covariance Matrix Adaptation-Evolution Strategy (CMA-ES) and Marine Predators Algorithm (MPA) algorithms. This is in terms of solving sets of unimodal and multimodal benchmark functions. Friedman and Holm–Bonferroni statistical tests have been utilized.

5.1. Utilized functions

Sets of six unimodal and six multimodal functions have been utilized. General simulation parameters are as tabulated in Table 4.

Fig. 10. Overall CFAPSO algorithm flowchart.

Essence of fuzzy-adaptive confidence and inertia weight parameters

Large values of personal confidence parameter as compared to social confidence parameter result in enhanced exploration. Large values of social confidence parameter as compared to personal confidence parameter result in enhanced exploitation and convergence. Consequently, it is imperative to have adaptive confidence parameters as opposed to the static values defined in the standard PSO algorithm. Good quality particles ought to be accorded high exploitation chances. Poor quality particles ought to be accorded high exploration chances. The proposed fuzzy logic scheme ties individual particle quality and iteration count to ideal parameter values.

A large inertia weight value is ideal in exploration stages and a smaller value in exploitation stages.

Essence of a particle culling procedure

To increase swarm diversity and consequently lower/ eliminate chances of stagnating at local optima, a particle culling procedure is developed. Poorly performing particles at half the number of maximum iterations are appropriately re-initialized.

Fig. 11. Essence of the modifications.

Table 4. General simulation parameters.

Dimension	40
Independent runs	50
Iterations per run	200
Processor	Intel Core i5-4300U at 1.9GHz, 2.49GHz
Memory	4 GB
Operating system	Windows, 64 bit
Simulation software	Matlab

Fig. 12. Function 1 (Sphere function).

Fig. 13. Function 2 (Schwefels function).

5.1.1. Unimodal functions

Unimodal functions are essential in evaluating exploitation capability of an algorithm. The utilized unimodal functions are herein briefly described. (See Figs. 12–17.)

Function 1

Formula:

$$f(\mathbf{x}) = \sum_{i=1}^n x_i^2 \quad (12)$$

Global minimum: 0

Search domain:

$$-100 \leq x_i \leq 100, 1 \leq i \leq n$$

Function 2

Formula:

$$f(\mathbf{x}) = \sum_{i=1}^n |x_i| + \prod_{i=1}^n |x_i| \quad (13)$$

Global minimum: 0

Search domain:

$$-100 \leq x_i \leq 100, 1 \leq i \leq n$$

Fig. 14. Function 3 (Quadric function).

Fig. 16. Function 5 (Rosenbrock function).

Fig. 15. Function 4.

Fig. 17. Function 6 (Step function).

Function 3

Formula:

$$f(\mathbf{x}) = \sum_{i=1}^n \left(\sum_{j=1}^n x_j \right)^2 \quad (14)$$

Global minimum: 0

Search domain:

$$-100 \leq x_i \leq 100, 1 \leq i \leq n$$

Function 4

Formula:

$$f(\mathbf{x}) = \text{Max}|x_i|, 1 \leq i \leq n \quad (15)$$

Global minimum: 0

Search domain:

$$-100 \leq x_i \leq 100, 1 \leq i \leq n$$

Function 5

Formula:

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \right] \quad (16)$$

Global minimum: 0

Search domain:

$$-30 \leq x_i \leq 30, 1 \leq i \leq n$$

Function 6

Formula:

$$f(\mathbf{x}) = \sum_{i=1}^n (|x_i + 0.5|)^2 \quad (17)$$

Global minimum: 0

Search domain:

$$-100 \leq x_i \leq 100, 1 \leq i \leq n$$

5.1.2. Multimodal functions

Multimodal functions are essential in evaluating exploration capability of an algorithm owing to the inherent large number of local optima. The utilized multimodal functions are herein briefly described. (See Figs. 18–23.)

Function 7

Formula:

$$f(\mathbf{x}) = \sum_{i=1}^n -x_i \sin(\sqrt{|x_i|}) \quad (18)$$

Global minimum: -418.98n

Search domain:

$$-500 \leq x_i \leq 500, 1 \leq i \leq n$$

Function 8

Formula:

$$f(\mathbf{x}) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)] \quad (19)$$

Global minimum: 0

Search domain:

$$-5.12 \leq x_i \leq 5.12, 1 \leq i \leq n$$

Fig. 18. Function 7.

Fig. 21. Function 10 (Griewank function).

Fig. 19. Function 8 (Rastrigin function).

Fig. 22. Function 11.

Fig. 20. Function 9 (Ackley function).

Function 9

Formula:

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

Global minimum: 0

Search domain:

$$-5 \leq x_i \leq 5, 1 \leq i \leq n$$

Function 10

Formula:

$$f(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (21)$$

Global minimum: 0

Search domain:

$$-600 \leq x_i \leq 600, 1 \leq i \leq n$$

Function 11

Formula:

$$f(\mathbf{x}) = \frac{\pi}{d} \{ 10 \sin(\pi y_1) + \sum_{i=1}^n (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \} + \sum_{i=1}^n u(x_i, 10, 100, 4) \quad (22)$$

Global minimum: 0

Search domain:

$$-50 \leq x_i \leq 50, 1 \leq i \leq n$$

Function 12

Formula:

$$f(\mathbf{x}) = 0.1 \{ \sin^2(3\pi x_1) + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \} + \sum_{i=1}^n u(x_i, 5, 100, 4) \quad (23)$$

Global minimum: 0

Search domain:

$$-50 \leq x_i \leq 50, 1 \leq i \leq n$$

Fig. 23. Function 12.

Fig. 24. Convergence plot: Function 1.

Fig. 25. Convergence plot: Function 2.

Fig. 26. Convergence plot: Function 3.

Fig. 27. Convergence plot: Function 4.

Fig. 28. Convergence plot: Function 5.

Fig. 29. Convergence plot: Function 6.

5.2. Performance analysis results

5.2.1. Exploitation test results (F1 to F6)

Figures 24 to 29 illustrate convergence curves obtained upon optimizing the unimodal test functions using the six algorithms under consideration. The presented convergence curves are the average outcome of the 50 independent runs. It is distinctly clear that the CFAPSO algorithm outperforms the other algorithms in optimizing Functions 1, 2, 4 and 6 (Figs. 24, 25, 27 and 29).

A Friedman test is performed to statistically prove/ disapprove that the six algorithms under study perform differently in optimizing the six unimodal functions. Table 5 gives the performance ranks of the six algorithms in optimizing the six unimodal functions. It can be deduced that the CFAPSO algorithm bears the best mean rank. Table 6 gives the overall Friedman test outcome. The null hypothesis is rejected, indicating that the performance of the six algorithms is different. A Holm-Bonferroni test is consequently carried out to pinpoint the algorithms bearing differing performance.

Tables 7 and 8 give the Holm-Bonferroni test results corresponding to the unimodal functions at significance levels 0.1 and 0.2 respectively.

The performance of the CFAPSO algorithm is found to be statistically different to that of GA and Basic PSO algorithms and similar to that of APSO, MPA, CMA-ES algorithms at a significance level of 0.1. The performance of the CFAPSO algorithm is statistically different to that of all other algorithms at a significance level of 0.2.

Table 5. Friedman ranking: Unimodal functions.

Func		CFAPSO	APSO	Basic PSO	GA	CMA-ES	MPA
F1	Mean	0.002	0.028	10.491	46.014	0.117	2.700
	Rank	1	2	5	6	3	4
F2	Mean	0.92	11.68	6.72	23.06	2.16	3.49
	Rank	1	5	4	6	2	3
F3	Mean	17882	36983	30712	28694	18248	13589
	Rank	2	6	5	4	3	1
F4	Mean	12.95	24.92	34.58	31.51	16.86	21.42
	Rank	1	4	6	5	2	3
F5	Mean	95.84	49.43	108.50	230.43	132.96	125.00
	Rank	2	1	3	6	5	4
F6	Mean	0.002	0.016	6.746	52.462	0.112	2.853
	Rank	1	2	5	6	3	4
$\sum$ ranks		8	20	28	33	18	19
Mean rank		1.33	3.33	4.67	5.50	3.00	3.17
$\sum$ ranks ²		64	400	784	1089	324	361

Table 6. Friedman test outcome: Unimodal functions.

alpha	0.05
Critical value	11.0704977
p-value	0.00306816
Friedman statistic	17.9047619
H ₀	All algorithms are identical
Reject Null?	YES

Fig. 30. Convergence plot: Function 7. **Fig. 31.** Convergence plot: Function 8. **Fig. 32.** Convergence plot: Function 9. **Fig. 33.** Convergence plot: Function 10. **Fig. 34.** Convergence plot: Function 11.

5.2.2. Exploration test results (F7 to F12)

Figures 30 to 35 illustrate convergence curves obtained upon optimizing the multimodal test functions using the six algorithms under consideration. The presented convergence curves are the average outcome of the 50 independent runs. It is distinctly clear that the CFAPSO algorithm outperforms the other algorithms in optimizing Functions 8, 9, 11 and 12 (Figs. 31, 32, 34 and 35).

A Friedman test is performed to statistically prove/ disapprove that the six algorithms under study perform differently in optimizing the six multimodal functions. Table 9 gives the performance ranks of the six algorithms in optimizing the six multimodal functions. It can be deduced that the CFAPSO algorithm bears the best mean rank. Table 10

gives the overall Friedman test outcome. The null hypothesis is rejected, indicating that the performance of the six algorithms is different. A Holm–Bonferroni test is consequently carried out to pinpoint the algorithms bearing differing performance.

Table 11 gives the Holm–Bonferroni test results corresponding to the multimodal functions at a significance level of 0.1. The performance

Table 7. Holm–Bonferroni test results: Unimodal functions.

Control: CFAPSO alpha = 0.1					
Data set	Z	P	Rank of P	Holm-B alpha	Reject Null H?
GA	3.86E + 00	1.15E-04	1	0.0200	YES
Basic PSO	3.09E + 00	2.03E-03	2	0.025	YES
APSO	1.85E + 00	6.41E-02	3	0.03	NO
MPA	1.70E + 00	8.96E-02	4	0.05	NO
CMA-ES	1.54E + 00	1.23E-01	5	0.1	NO

Table 8. Holm–Bonferroni test results: Unimodal functions.

Control: CFAPSO alpha = 0.2					
Data set	Z	P	Rank of P	Holm-B alpha	Reject Null H?
GA	3.86E + 00	1.15E-04	1	0.04	YES
Basic PSO	3.09E + 00	2.03E-03	2	0.05	YES
APSO	1.85E + 00	6.41E-02	3	0.0667	YES
MPA	1.70E + 00	8.96E-02	4	0.1	YES
CMA-ES	1.54E + 00	1.23E-01	5	0.2	YES

Table 9. Friedman ranking: Multimodal functions.

Func		CFAPSO	APSO	Basic PSO	GA	CMA-ES	MPA
F7	Mean	-11752	-12023	-11731	-11416	-11507	-10982
	Rank	2	1	3	5	4	6
F8	Mean	120.60	157.69	167.12	170.00	125.80	129.80
	Rank	1	4	5	6	2	3
F9	Mean	0.05	0.22	0.44	2.13	1.45	1.22
	Rank	1	2	3	6	5	4
F10	Mean	0.03	1.35	1.01	0.01	0.70	0.15
	Rank	2	6	5	1	4	3
F11	Mean	0.94	4.69	13.37	1.51	6.23	2.04
	Rank	1	4	6	2	5	3
F12	Mean	0.33	3.44	7.02	810.32	45.12	25.04
	Rank	1	2	3	6	5	4
$\sum$ ranks		8	19	25	26	25	23
Mean		1.33	3.17	4.17	4.33	4.17	3.83
$\sum$ ranks ²		64	361	625	676	625	529

Fig. 35. Convergence plot: Function 12. **Fig. 36.** Average computation time.**Table 10.** Friedman test outcome: Multimodal functions.

alpha	0.05
Critical value	11.0704977
p-value	0.04861997
Friedman statistic	11.1428571
H ₀	All algorithms are identical
Reject Null?	YES

of the CFAPSO algorithm is statistically different to that of all other algorithms.

5.3. Computational complexity analysis

Herein, time complexity analysis of the six algorithms under comparison is presented.

5.3.1. Average computation time

Table 12 gives the six algorithms' average computation time for a single iteration. The time has been calculated with respect to the twelve (unimodal and multimodal) optimization problems. The algorithms' rank in terms of computation time has been given. The order (ascending) of performance is: Basic PSO, APSO, CFAPSO, MPA, GA and CMA-ES. The Basic PSO algorithm yields the lowest computation time. This can be directly attributed to the generally simple structure of the algorithm. It is worth noting that a metaheuristic algorithm computation complexity in general doesn't relate to the overall algorithm quality in terms of solution generation performance. Despite the good time complexity performance, the Basic PSO algorithm has a tendency of premature convergence.

Fig. 36 is a graphical illustration of the computation time data given in Table 12.

Table 11. Holm–Bonferroni test results: Multimodal functions.

Control: CFAPSO alpha = 0.1					
Data set	Z	P	Rank of P	Holm-B alpha	Reject Null H?
GA	2.78E+00	5.48E-03	1	0.02	YES
Basic PSO	2.62E+00	8.71E-03	2.5	0.028571429	YES
CMA-ES	2.62E+00	8.71E-03	2.5	0.028571429	YES
MPA	2.31E+00	2.06E-02	4	0.05	YES
APSO	1.70E+00	8.96E-02	5	0.1	YES

Table 12. Average computation time (per iteration).

Algorithm	Average computation time (ms)	Rank
CFAPSO	4.553	3
APSO	4.385	2
Basic PSO	4.213	1
GA	4.758	5
CMA-ES	4.885	6
MPA	4.626	4

Fig. 37. Average computation time.**Table 13.** Computation time: Function 8, at function val 200.

Algorithm	No. of Iter	Time/ Iter (ms)	Total time (ms)	Rank
CFAPSO	106	4.530	480.180	1
APSO	126	4.370	550.620	5
Basic PSO	127	4.170	529.590	4
GA	153	4.750	726.750	6
CMA-ES	106	4.860	515.160	2
MPA	114	4.620	526.680	3

Fig. 37 illustrates the computation time taken for a given number of algorithm iterations.

5.3.2. Average time to get a certain result in a specific problem (Function 8)

A consideration is made on the total computation time taken in optimizing Function 8. In particular, the time taken to obtain an outcome of $Function_8 = 200$ is considered. The computation time is as tabulated in Table 13. In this case, the CFAPSO algorithm outperforms the other algorithms. The CFAPSO algorithm requires fewer iterations

Fig. 38 is a graphical illustration of the computation time data given in Table 13.

6. Collaborative beamforming using the CFAPSO algorithm

Herein, the performance of the developed CFAPSO algorithm in CBF has been analyzed. Performance comparison has been done against other PSO algorithm variants (*linearly-adaptive PSO*, *basic PSO*) and GA.

6.1. Methodology

The general methodology followed is as per the listing below:

Fig. 38. Average computation time.**Table 14.** General parameters (apply to all utilized algorithms).

Parameter	Value
Population/ Swarm size	30
Swarm initialization	Random
Upper phase bound	2π radians
Lower phase bound	-2π radians
Upper amplitude bound	1
Lower amplitude bound	-1
Iterations	60

- The model developed in Section 2 is utilized as the algorithm performance comparison basis (in terms of solving a beamsteering problem).
- The algorithms are run over identical (60) iterations. A huge chunk of computation time is spent handling cost function evaluation as opposed to handling algorithm intricacies. As such, an iteration count comparison basis offers an outcome more or less similar to that of a time-based comparison.
- The algorithms' "particle swarms"/ "populations" are identically initialized to allow for fair comparison.
- Owing to the stochastic nature of the algorithms under study, average outcomes emanating from 50 independent runs are utilized as the comparison basis.
- Results are presented qualitatively (in the form of radiation pattern plots) and quantitatively (radiation power and beamsteering accuracy data).
- Appropriate statistical analysis (*ANalysis Of VAriance (ANOVA) comparison test alongside Tukey-Kramer post-hoc analysis*) is performed on the obtained data.

The utilized performance metrics are: Beamforming accuracy and the value of the normalized power in the desired direction. Utilized GA algorithm parameters are: Elite Count = 10 and Crossover Fraction = 2/3. Other algorithm parameters are as per the tabulations given in Tables 14 and 15.

6.2. General problem formulation

A set of ten nodes is configured in a planar manner as per the illustration given in Fig. 39. The nodes are randomly distributed over

Table 15. Basic PSO, APSO and CFAPSO algorithms' parameters.

Parameter	Value		
	Basic PSO	APSO	CFAPSO
Inertia weight	Lin. ad. (0.9 to 0.4)	Lin. ad. (0.9 to 0.4)	Fuz. ad. (0.9 to 0.4)
Personal conf.	Fixed at 2.4	Lin. ad. (2.6 to 2.2)	Fuz. ad. (2.6 to 2.2)
Social conf.	Fixed at 2.2	Lin. ad. (2 to 2.4)	Fuz. ad. (2 to 2.4)
Neighbourhoods	4	4	4

Fig. 39. Node layout: A random and planar configuration over an area of unit radius (wherein the radius is wavelength-normalized).

a coverage area of unit radius (wherein the radius is wavelength-normalized). A wavelength of 0.125 m is utilized assuming an operation frequency of 2.4 GHz.

The Sink location from the perspective of the collaborating nodes is -120 degrees azimuth, 50 degrees elevation (in the far-field region).

The objective function to be optimized is as per Eq. (24).

$$\text{maximize} \quad |AF_{\phi_0, \theta_0}(\mathbf{w})|^2 \approx \left| \sum_{k=1}^{10} w_k e^{j2\pi[-\tilde{r}_k \sin \theta_0 \cos(\phi_0 - \psi_k)]} \right|^2 \quad (24)$$

Where:

- (ϕ_0, θ_0) is the sink direction
- $|AF_{\phi_0, \theta_0}|^2$ is the power observed in the direction of the sink.
- w_k are complex beamsteering weights (nodes' transmission weights).

The objective function as presented in Eq. (24) can be maximized by optimally adjusting the nodes' transmission complex weights (w_k). The complex weights can be expressed as per Eq. (25) where A_k is the amplitude factor and B_k is the phase factor.

$$w_k = A_k \exp^{jB_k} \quad (25)$$

The algorithms under study have been configured to obtain optimal complex weights (as per the guideline given in Section 4.5).

6.3. Results analysis/ discussion

In this section, the results obtained in the algorithm comparison process are laid out and analyzed.

6.3.1. Section A: normalized power patterns in the form of contour plots

The normalized power pattern corresponding to the initial node weights is illustrated in Figure Fig. 40.

The power pattern presented in Figure Fig. 40 reflects the low average amplitude of initial node weights (randomly distributed within the limits [-1,1]).

Normalized power patterns in the form of contour plots corresponding to CBF using the algorithms under study are given in Figs. 41–45. The presented power patterns are statistical averages corresponding to power patterns derived over 50 independent algorithm runs.

Fig. 40. Normalized power pattern: initial node weights. **Fig. 41.** Average normalized power pattern: CFAPSO algorithm. **Fig. 42.** Average normalized power pattern: APSO algorithm. **Fig. 43.** Average normalized power pattern: Basic PSO algorithm. **Fig. 44.** Average normalized power pattern: GA algorithm.

As per Figs. 41–45, high radiation power is directed towards the sink direction in all cases under study as expected. A relatively high sidelobe appears in the direction (80 deg. azimuth, 70 deg. elevation). It can be qualitatively deduced (albeit vaguely) that the CFAPSO algorithm outperforms the other algorithms under consideration. The CFAPSO algorithm power pattern roughly matches the conventional beamsteering power pattern.

Fig. 45. Normalized power pattern: Conventional CBF.

Fig. 46. Average objective function value against algorithm iterations.

Fig. 47. Azimuth cut of the normalized power pattern (at the elevation angle 50 degrees).

6.3.2. Section B: comparative plots

Fig. 46 comparatively illustrates the evolution of the CBF cost function (average outcome of 50 independent runs for each and every algorithm under study). It can be clearly deciphered that the CFAPSO algorithm outperforms the other algorithms. The common starting point reflects the aspect of a common initial seed among the algorithms under study.

Figs. 47 and 48 comparatively illustrate azimuth and elevation cuts of the normalized power pattern. Normalization is with respect to the highest power level achieved upon CBF with the algorithms under study (specifically the highest power level corresponding to conventional CBF). It can be clearly deciphered that the CFAPSO algorithm outperforms the other algorithms under study in terms of the power radiated towards the desired direction. The pattern corresponding to CBF using the CFAPSO algorithm almost matches that of conventional CBF.

6.3.3. Section C: quantified results

The normalized power observed in the desired direction upon CBF is as tabulated in Table 16. The tabulated data corresponds to the average outcome of 50 independent runs.

A statistical analysis is performed to establish that indeed the differences observed in the normalized power data are statistically significant (Tables 17 and 18).

Analysis of variance results corresponding to *normalized power in the desired direction* are tabulated in Table 17. The low P value (less than

Fig. 48. Elevation cut of the normalized power pattern (at the azimuth angle -120 degrees).

Table 16. Normalized power observed in the desired direction (averaged over 50 independent runs).

Algorithm	Power in desired direction (normalized)	
	Average value	SD
CFAPSO	0.963415546	0.066884801
APSO	0.871229978	0.083915913
Standard PSO	0.745359997	0.122197342
GA	0.645317791	0.129364561
Conventional	0.99998319	0

Table 17. Analysis of variance: Normalized power in the desired direction.

Source	SS	df	MS	F	P
Between	4.426791	4	1.106698	128.140935	0.000000
Within	2.115959	245	0.008637		
Total	6.542750	249	0.026276		

0.05) is indicative of significant unitary or multiple differences in the analyzed data-set.

The exact statistically significant differences in the data given in Table 16 are brought out in the Tukey-Kramer post-hoc analysis results presented in Table 18. It can be deduced that the performance of the CFAPSO algorithm statistically matches that of the conventional beam-steering approach.

The absolute deviation of the direction of peak power from the desired direction upon CBF is as tabulated in Table 19. The tabulated data corresponds to the average outcome of 50 independent runs.

Analysis of variance results corresponding to *absolute deviation of the direction of peak power from the desired direction* are tabulated in Tables 20 and 21. The low P values (less than 0.05) are indicative of significant unitary or multiple differences in the analyzed data-sets.

The exact statistically significant differences in the data given in Table 19 are brought out in the Tukey-Kramer post-hoc analysis results presented in Tables 22 and 23. Again, it can be deduced that the performance of the CFAPSO algorithm statistically matches that of the conventional beamsteering approach.

Tabulated in Table 24 are the percentages of direct “hits” from the perspective of perfect alignment between the direction of maximum power and the intended radiation direction. It can be deduced that the CFAPSO algorithm outperforms the other algorithms under study.

7. Conclusion

In this paper, the PSO algorithm has been modified and applied in CBF in the special case of a planar random arrangement of sensor nodes with the sink at an elevated plane. The modifications done on the PSO algorithm entail use of fuzzy-adaptive confidence and inertia weight parameters alongside a particle culling procedure. The modified PSO algorithm has been christened CFAPSO algorithm. Comparisons against an adaptive PSO algorithm variant (APSO), basic PSO, MPA, CMA-ES and the GA algorithms have established the superiority of the developed

Table 18. Tukey-Kramer comparison test: Normalized power in desired direction.

Comparison	Absolute Difference	Std. Error of Difference	Critical Range	Results
CFAPSO to APSP	0.092186	0.01314273	0.0511	Means are different
CFAPSO to SPSP	0.218056			Means are different
CFAPSO to SGA	0.318098			Means are different
CFAPSO to Conv	0.036583			Means are not different
APSP to SPSP	0.12587			Means are different
APSP to SGA	0.225912			Means are different
APSP to Conv	0.128768			Means are different
SPSP to SGA	0.100042			Means are different
SPSP to Conv	0.254638			Means are different
SGA to Conv	0.354681			Means are different

Table 19. Absolute deviation of the direction of peak power from the desired direction.

Algorithm	Azimuth angle		Elevation angle	
	Absolute error	SD	Absolute error	SD
CFAPSO	0.46	0.734291273	0.48	0.677329691
APSP	0.94	1.01840211	1.22	1.015994536
Standard PSO	1.62	1.047640692	1.9	1.474269103
GA	1.92	1.536494818	2	1.761261144
Conventional	0	0	0	0

Table 20. Analysis of variance: Azimuth angle deviation.

Source	SS	df	MS	F	P
Between	126.264000	4	31.566000	31.348480	0.000000
Within	246.700000	245	1.006939		
Total	372.964000	249	1.497847		

Table 21. Analysis of variance: Elevation angle deviation.

Source	SS	df	MS	F	P
Between	152.840000	4	38.210000	28.234558	0.000000
Within	331.560000	245	1.353306		
Total	484.400000	249	1.945382		

CFAPSO algorithm in solving a set of standard unimodal and multimodal functions. The general performance of the CFAPSO algorithm in CBF has been analyzed. As far as CBF is concerned, the CFAPSO algorithm is found to generate a beamsteering outcome nearly identical to that of conventional beamsteering. Despite yielding ideal beamsteering outcomes, it is noteworthy that conventional beamsteering is inapplicable to beam-pattern optimization with respect to processes such as sidelobe minimization, nulling among others. The developed CFAPSO algorithm befits utilization in beamsteering alongside beam-pattern optimization.

Declarations

Author contribution statement

Robert Macharia Maina: Conceived and designed the experiments; Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper. **Philip Kibet Lang'at;** **Peter Kamita Kihato:** Conceived and designed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data.

Funding statement

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data availability statement

Data will be made available on request.

Declaration of interests statement

The authors declare no conflict of interest.

Additional information

No additional information is available for this paper.

Acknowledgements

This work was supported by Institute of Basic Science, Technology and Innovation, Pan African University, Nairobi, Kenya.

The authors would also like to thank the editors and anonymous reviewers for providing insightful suggestions and comments to improve the quality of research paper.

References

- [1] D. Kandris, C. Nakas, D. Vomvas, G. Koulouras, Applications of wireless sensor networks: an up-to-date survey, *Appl. Syst. Innov.* 3 (1) (2020) 14.
- [2] O.I. Khalaf, B.M. Sabbar, An overview on wireless sensor networks and finding optimal location of nodes, *Period. Eng. Nat. Sci.* 7 (3) (2019) 1096–1101.
- [3] M. Abdulkareem, K. Samsudin, F.Z. Rokhani, M.F.A. Rasid, Wireless sensor network for structural health monitoring: a contemporary review of technologies, challenges, and future direction, *Struct. Health Monit.* 19 (3) (2020) 693–735.
- [4] C.S. Raghavendra, K.M. Sivalingam, T. Znati, *Wireless Sensor Networks*, Springer, 2006.
- [5] S. Khan, A.K. Pathan, N.A. Alrajeh, *Wireless Sensor Networks: Current Status and Future Trends*, CRC Press, 2016.
- [6] K. Sohraby, D. Minoli, T. Znati, *Wireless Sensor Networks: Technology, Protocols, and Applications*, John Wiley and Sons Inc., 2007.
- [7] H. Ochiai, P. Mitran, H.V. Poor, V. Tarokh, Collaborative beamforming for distributed wireless ad hoc sensor networks, *IEEE Trans. Signal Process.* 53 (11) (2005) 4110–4124.
- [8] G. Sun, Y. Liu, Z. Chen, A. Wang, Y. Zhang, D. Tian, V.C.M. Leung, Energy efficient collaborative beamforming for reducing sidelobe in wireless sensor networks, *IEEE Trans. Mob. Comput.* 20 (3) (2021) 965–982.
- [9] S. Liang, Z. Fang, G. Sun, Y. Liu, X. Zhao, G. Qu, Y. Zhang, V.C.M. Leung, Jssa: joint sidelobe suppression approach for collaborative beamforming in wireless sensor networks, *IEEE Access* 7 (2019) 151803–151817.
- [10] S. Liang, Z. Fang, G. Sun, Y. Liu, G. Qu, S. Jayaprakasam, Y. Zhang, A joint optimization approach for distributed collaborative beamforming in mobile wireless sensor networks, *Ad Hoc Netw.* 106 (2020) 102216.
- [11] G. Sun, X. Zhao, G. Shen, Y. Liu, A. Wang, S. Jayaprakasam, Y. Zhang, V.C.M. Leung, Improving performance of distributed collaborative beamforming in mobile wireless sensor networks: a multiobjective optimization method, *IEEE Int. Things J.* 7 (8) (2020) 6787–6801.
- [12] X. Bao, H. Liang, L. Han, A novel node selection algorithm for collaborative beamforming in wireless sensor networks, in: 2018 IEEE International Conference on Internet of Things, IEEE, 2018, pp. 345–349.

Table 22. Tukey-Kramer comparison test: Azimuth angle deviation.

Comparison	Absolute Difference	Std. Error of Difference	Critical Range	Results
CFAPSO to APSP	0.48	0.14191115	0.552	Means are not different
CFAPSO to SPSP	1.16			Means are different
CFAPSO to SGA	1.46			Means are different
CFAPSO to Conv	0.46			Means are not different
APSP to SPSP	0.68			Means are different
APSP to SGA	0.98			Means are different
APSP to Conv	0.94			Means are different
SPSP to SGA	0.3			Means are not different
SPSP to Conv	1.62			Means are different
SGA to Conv	1.92			Means are different

Table 23. Tukey-Kramer comparison test: Elevation angle deviation.

Comparison	Absolute Difference	Std. Error of Difference	Critical Range	Results
CFAPSO to APSP	0.74	0.16451785	0.64	Means are different
CFAPSO to SPSP	1.42			Means are different
CFAPSO to SGA	1.52			Means are different
CFAPSO to Conv	0.48			Means are not different
APSP to SPSP	0.68			Means are different
APSP to SGA	0.78			Means are different
APSP to Conv	1.22			Means are different
SPSP to SGA	0.1			Means are not different
SPSP to Conv	1.9			Means are different
SGA to Conv	2			Means are different

Table 24. Percentage accuracy in beamsteering.

	Azimuth	Elevation	Both Az. and El.
CFAPSO	62	60	42
APSP	36	24	10
Standard PSO	22	19	8
GA	14	16	6
Conventional	100	100	100

- [13] S. Jayaprakasam, S.K.A. Rahim, C.Y. Leow, Distributed and collaborative beamforming in wireless sensor networks: classifications, trends, and research directions, *IEEE Commun. Surv. Tutor.* 19 (4) (2017) 2092–2116.
- [14] H. Chen, D.L. Fan, L. Fang, W. Huang, J. Huang, C. Cao, L. Yang, Y. He, L. Zeng, Particle swarm optimization algorithm with mutation operator for particle filter noise reduction in mechanical fault diagnosis, *Int. J. Pattern Recognit. Artif. Intell.* 34 (10) (2020) 2058012.
- [15] L.M. Abualigah, A.T. Khader, E.S. Hanandeh, A new feature selection method to improve the document clustering using particle swarm optimization algorithm, *J. Comput. Sci.* 25 (2018) 456–466.
- [16] F. Wang, H. Zhang, K. Li, Z. Lin, J. Yang, X.-L. Shen, A hybrid particle swarm optimization algorithm using adaptive learning strategy, *Inf. Sci.* 436 (2018) 162–177.
- [17] Z. Cui, J. Zhang, D. Wu, X. Cai, H. Wang, W. Zhang, J. Chen, Hybrid many-objective particle swarm optimization algorithm for green coal production problem, *Inf. Sci.* 518 (2020) 256–271.
- [18] I.B. Aydi, A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems, *Appl. Soft Comput.* 66 (2018) 232–249.
- [19] Y. Cao, H. Zhang, W. Li, M. Zhou, Y. Zhang, W.A. Chaovalitwongse, Comprehensive learning particle swarm optimization algorithm with local search for multimodal functions, *IEEE Trans. Evol. Comput.* 23 (4) (2018) 718–731.
- [20] Z. Xin-gang, L. Ji, M. Jin, Z. Ying, An improved quantum particle swarm optimization algorithm for environmental economic dispatch, *Expert Syst. Appl.* 152 (2020) 113370.
- [21] S. Nadweh, O. Khaddam, G. Hayek, B. Atieh, H.H. Alhelou, Steady state analysis of modern industrial variable speed drive systems using controllers adjusted via grey wolf algorithm & particle swarm optimization, *Heliyon* 6 (11) (2020) e05438.
- [22] O. Tarkhaneh, H. Shen, Training of feedforward neural networks for data classification using hybrid particle swarm optimization, mantegna Lévy flight and neighborhood search, *Heliyon* 5 (4) (2019) e01275.
- [23] K.R. Harrison, A.P. Engelbrecht, B.M. Ombuki-Berman, An adaptive particle swarm optimization algorithm based on optimal parameter regions, in: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2017, pp. 1–8.
- [24] Y. Marinakis, M. Marinaki, A. Migdalas, A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows, *Inf. Sci.* 481 (2019) 311–329.
- [25] M.A. Mellal, E. Zio, An adaptive particle swarm optimization method for multi-objective system reliability optimization, *Proc. Inst. Mech. Eng., Part O: J. Risk Reliab.* 233 (6) (2019) 990–1001.
- [26] C.Y. Tazibt, M. Bekhti, T. Djamah, N. Achir, K. Boussetta, Wireless sensor network clustering for uav-based data gathering, in: 2017 Wireless Days, 2017, pp. 245–247.
- [27] R. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: Proceedings of IEEE Congress on Evolutionary Computation, 2000, pp. 84–88.
- [28] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942–1948.
- [29] T. Weise, Global Optimization Algorithms: Theory and Applications, 3rd edition, University of Kassel, Kassel, 2009.
- [30] Maurice-Clerc, Particle Swarm Optimization, 1st edition, ISTE Ltd, London, UK, 2006.
- [31] P. Erdogmus, Particle Swarm Optimization with Applications, 1st edition, InTechOpen, 2018.
- [32] M. Affenzeller, S. Wagner, S. Winkler, A. Beham, Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications, 1st edition, Chapman and Hall, 2018.
- [33] S.N. Shahbazova, M. Sugeno, J. Kacprzyk, Recent Developments in Fuzzy Logic and Fuzzy Sets, Springer, 2020.
- [34] V. Sharma, A.K. Gupta, A comprehensive study of fuzzy logic, *Int. J. Adv. Res. Comput. Sci. Softw. Eng.* 3 (2) (2013) 398–403.