Computers, Materials & Continua
DOI: 10.32604/cmc.2023.032183
Article

Tech Science Press

Hybrid Global Optimization Algorithm for Feature Selection

Ahmad Taher Azar[1,2,*], Zafar Iqbal Khan[2], Syed Umar Amin[2] and Khaled M. Fouad[1,3]

[1]Faculty of Computers and Artificial Intelligence, Benha University, Benha, 13511, Egypt
[2]College of Computer and Information Sciences, Prince Sultan University, Riyadh, 11586, Saudi Arabia
[3]Faculty of Information Technology and Computer Science, Nile University, Sheikh Zaid, Egypt
*Corresponding Author: Ahmad Taher Azar. Emails: ahmad.azar@fci.bu.edu.eg, aazar@psu.edu.sa
Received: 10 May 2022; Accepted: 21 June 2022

Abstract: This paper proposes Parallelized Linear Time-Variant Acceleration Coefficients and Inertial Weight of Particle Swarm Optimization algorithm (PLTVACIW-PSO). Its designed has introduced the benefits of Parallel computing into the combined power of TVAC (Time-Variant Acceleration Coefficients) and IW (Inertial Weight). Proposed algorithm has been tested against linear, non-linear, traditional, and multiswarm based optimization algorithms. An experimental study is performed in two stages to assess the proposed PLTVACIW-PSO. Phase I uses 12 recognized Standard Benchmarks methods to evaluate the comparative performance of the proposed PLTVACIW-PSO *vs.* IW based Particle Swarm Optimization (PSO) algorithms, TVAC based PSO algorithms, traditional PSO, Genetic algorithms (GA), Differential evolution (DE), and, finally, Flower Pollination (FP) algorithms. In phase II, the proposed PLTVACIW-PSO uses the same 12 known Benchmark functions to test its performance against the BAT (BA) and Multi-Swarm BAT algorithms. In phase III, the proposed PLTVACIW-PSO is employed to augment the feature selection problem for medical datasets. This experimental study shows that the planned PLTVACIW-PSO outpaces the performances of other comparable algorithms. Outcomes from the experiments shows that the PLTVACIW-PSO is capable of outlining a feature subset that is capable of enhancing the classification efficiency and gives the minimal subset of the core features.

Keywords: Particle swarm optimization (PSO); time-variant acceleration coefficients (TVAC); genetic algorithms; differential evolution; feature selection; medical data

1 Introduction

The Concept of swarm intelligence (SI) principle, is highly inspired by the recent advancement in the field of Neuroscience and the Behavioral Science, commonly known as intelligent paradigm in Intelligence Computational domain, to solve the optimization issues of various problems in absence of any global models [1,2]. As a result, the swarm concept, inspired by the mutual attitudes of gregarious

animals (like a herd of animals, birds, fishes, ants, cats, and fireflies), is used to discuss intelligent agents in a distributed system [3]. A swarm is an aggregation of identical objects which may accomplish some task, may interact among themselves in a predefined manner, may react to surrounding environments, in absence of any central governing entity. Algorithms which are based upon this kind of behavior, imitate natural appearing phenomenon, an efficient and viable solution to several complex problems [4]. In these circumstances, Particle Swarm Optimization or better known as PSO, open horizons for meta-heuristic global optimization techniques evolved from concepts of swarm intelligence [5].

Kennedy et al., proposed a particle swarm optimization algorithm, based on meta-heuristics evolved from swarm intelligence methodologies, it can emulate cordial movement patterns of flocks of birds and flying patterns of birds, it has capabilities for intercommunication among group members, to assist them in decision making processes, in a coherent synchronized manner [5].

Swarms are simulated, as if flocks are out in search of their food, any particular individual in the flock, determines its relative speed and position by calculating these two attributes of its neighbors speed and positions among the flock members. During Optimization of solution space, a swarm particle in a PSO changes its positions in a multidimensional space. This phenomenon reconciles location of particle under solution space for the problem being optimized. These particles are controlled by a tradeoff of memory between the group and the individuals. As stated in [5,6], the main idea of particle swarm optimization was influenced by bird flocks hunting food sources. Previous research has demonstrated that optimization using the PSO method yields accurate results. In a PSO algorithm, a particle's movement is determined by its location and velocity.

Iteratively, the particle velocity determines its route. The particle velocity is determined by three key factors. The first component, a social component, tries to dominate the best position for all particles I during a single iteration or to keep the global best position till the next iteration, which is termed the current global best position (gbesti), where i is the particle's index. The second component, a cognitive component, tries to dominate each particle I in the swarm individually, or the personal best for a given particle, until the present iteration, dubbed the current best for particle i. (pbesti). The third component, a momentum component, determines the influence of each particle's past velocity and is considered a modification of the original PSO described in [7]. The PSO's position and velocity are determined by Eqs. (1) and (2).

$$v_i(t+1) = wv_i(t) + ((c_1r_1) \times pbest_i(t) - x_i(t)) + ((c_2r_2) \times gbest_i(t) - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

where xi(t) denotes the location of a particle i at time t, and vi(t) denotes the velocity of a particle i at time t. w is the inertial weight that prevents particles from drifting away from their optimal places without attaining convergence; The time-variant acceleration coefficients (TVAC) c1 and c2 strive to alter the weight of two components, social and cognitive, in order to dominate personal and local optimal attitudes [8]; pbesti(t) is the best-known position of particle i at a specific time t; gbesti(t) is the best-known position of a neighbor particle to i; and r_1 and r_2 are independent random values in a range of [0, 1]. The PSO algorithm begins its computational steps to determine the current position of each particle, and then checks if this position is optimal to attain the best position. The next step involves computing the velocity of the particle using Eq. (1) and then updating the position of the particle using Eq. (2). Initially, when a cognitive component is large and a social component is small, the particles move around the search space instead of moving across the best population. Furthermore, when the cognitive component is low, and the social component is large, the particles converge at the global optima of the latter part of optimization.

The proposed Parallelized Linear Time-Variant Acceleration Coefficients and Inertial Weight of PSO (PLTVACIW-PSO) algorithm, based on a hybridization of the linearity of time-variant acceleration coefficients and a linear decrease of the inertial weight associated with the parallel processing of PSO fitness evaluation, will be used in this paper. The proposed algorithm will be used to optimize 12 known benchmark functions, varying in their properties and scales. The optimization of these functions tends to minimize the errors generated in the sequential iterations of the proposed PLTVACIW-PSO *vs.* other PSO TVAC algorithms, PSO IW algorithms, GA and DE algorithms, and, lastly, BAT and Multi-swarm BAT algorithms. The four categories mentioned before are shown in Tab. 1. The first category is the "IW family", which consists of four parallelized and non-parallelized inertial weight-based PSO algorithms called PLIW-PSO, LIW-PSO, PNLIW-PSO, and NLIW-PSO. The second category is the "TVAC family", which consists of four parallelized and non-parallelized TVAC based PSO algorithms called PLTV-PSO, LTV-PSO, PNLTV-PSO, and NLTV-PSO. The third category includes traditional algorithms: PSO, the Differential Evolution algorithm (DE), the Genetic algorithm (GA), and the Flower Pollination Algorithm (FPA). Finally, two bio-inspired algorithms are compared: the BAT and Multi swarm-BAT algorithms.

Table 1: Related algorithms

Algorithm name	Short name	Category	Referenced or implemented	Description
Linear Inertial Weight PSO	LIW-PSO	IW family	[9]	Inertial weight w varies linearly over time.
Parallelized Linear Inertial Weight PSO	PLIW-PSO		Implemented	Parallelized version of LIW-PSO
Non-Linear Inertial Weight PSO	NLIW-PSO		[10]	Inertial weight w varies non-linearly over time.
Parallelized Non-Linear Inertial Weight PSO	PNLIW-PSO		Implemented	Parallelized version of NLIW-PSO
Linear Time-Variant PSO	LTV-PSO	TVAC family	[8]	PSO acceleration variables c_1 and c_2 linearly vary over time.
Parallelized Linear Time Variant PSO	PLTV-PSO		[11]	Parallelized version of LTV-PSO
Non-Linear Time-Variant PSO	NLTV-PSO		[12]	PSO acceleration variables c_1 and c_2 non-linearly vary overtime.

(Continued)

Table 1: Continued

Algorithm name	Short name	Category	Referenced or implemented	Description
Parallelized Non-Linear Time Variant PSO	PNLTV-PSO		[11]	Parallelized version of NLTV-PSO
Particle Swarm Optimization	PSO	Traditional algorithms	[5]	A nature-inspired algorithm based on the behavior of bird swarms and fish schools.
Differential Evolution	DE		[13]	A modified version of genetic algorithms
Genetic Algorithms	GA		[14]	A nature-inspired optimization algorithm based on human genes and their crossover and mutation
Flower Pollination Algorithm	FPA		[15]	A nature-inspired optimization algorithm based on flower pollination behavior.
Bat	BAT	Bat algorithms	[16]	A bio-inspired stochastic algorithm based on a swarm of bats
Multi-swarm BAT	MBAT		[16]	A bio-inspired stochastic algorithm based on multiple swarms of bats.

2 Preliminaries

The PSO algorithm utilises meta-parameters, which controls the swarm actions for efficient optimization and it effective in enhancing searchability of a particle in the swarm. The convergence characteristics parameters of PSO algorithm are dependent upon controlling parameters, hence impact on controlling parameters also impacts convergence characteristics. Therefore, the functions of these meta-parameters and their influence on the conclusive results are crucially importance for designing an efficient optimization algorithm [7].

2.1 Swarm Initialization

The random position of each particle is the initialization step in the PSO algorithm, which starts inside an iteration to keep searching for optimal solutions. The velocities and positions for each particle are determined in every iteration until the final iteration or until any stopping criteria are provided.

Non-uniformly disseminated introductory particles impact the minimal stability properties of the PSO algorithm. Furthermore, the convergence of the particle's velocity in the swarm is dependent on the initial population [17].

2.2 Parameter Determination

The PSO algorithm's control parameters are the inertial weight and acceleration coefficients. The correct setting of these parameters can have an impact on the PSO's convergence. The PSO algorithm's cognitive and social components are influenced by the acceleration coefficients c1 and c2 [18,19]. The acceleration coefficients [20] refer to the speed at which the particles are moved to the swarm's most precise location and superior position. They tweak a particle's distance and timing to get it into the best possible spot. If the acceleration coefficients are excessively great, the particle may go past the right place by accident. If the acceleration coefficients are too little, the particle will not be able to reach its intended position. This maximum particle's velocity (Vmax) identifies the intention, which controls the areas between the current particle's position and the target particle's position [21]. If Vmax value is too high, the particles could miss well-known locations. However, if Vmax is too small, the particles may not reach very distant places. Therefore, the determination of Vmax should be tuned by using a weight factor, which should be added to the PSO equation (Eq. (1)) [20]. This weight is understood as inertial weight (w), which aims to dominate the speed of the arrival at the best location during the final iteration. An increase in the value of w enhances global searchability, whereas a decrease in the value of w enhances partial searchability.

2.3 Parameters Enhancements

To increase the performance of the PSO algorithm, the TVAC was upgraded [8,11]. TVAC may raise or reduce the swarm's cognitive and social behavior in a linear fashion. In other words, allowing the swarm to travel the whole search area by enhancing cognitive aspects and minimizing social factors in the early phases of the optimization process. The convergence of all particles toward the global optima at the conclusion of the optimization process is improved by reducing cognitive aspects and enhancing social ones. Nonlinear inertial weight variation might be used as a benchmark for nonlinearly enhancing acceleration coefficients [10]. Instead of manually establishing the inertial weight parameters, the speed of convergence of the relevant particles in the swarm might be enhanced. The following equations indicate improvements to c1 and c2:

$$c_1 = \left(c_{1f} - c_{1i}\right) \times \left(\frac{iter}{MAXITER}\right) + c_{1i} \tag{3}$$

$$c_2 = \left(c_{2f} - c_{2i}\right) \times \left(\frac{iter}{MAXITER}\right) + c_{2i} \tag{4}$$

where c1i, c1f and c2i, c2f are the initial and final values of c1 and c2, respectively; iter is the currently running iteration; and MAXITER is the maximum number of iterations the algorithm will perform.

Another strategy for enhancing the swarm's particle convergence rate is linear inertial weight w adaptation [9]. The upper and lower inertial weight limits are established, and a new inertial weight value is computed and evaluated in each iteration to adjust the swarm to the optimal location in the search space. The improvement in inertial weight w is seen in the following equation:

$$w = (w_{\max}) - \left(\frac{iter \times (w_{\max} - w_{\min})}{MAXITER}\right) \tag{5}$$

where w1 and w2 are the initial and final values of w, respectively.

2.4 Parallel Processing of the PSO Algorithm

The PSO algorithm's execution time was reduced and the fitness assessment of individual particles in the swarm was reduced by parallel processing of objective functions across multiples of the independent machine in the form of a cluster [22]. Before calculating a new velocity and location in subsequent rounds of the PSO algorithm, all participating machines must achieve global synchronization to ensure that each particle's fitness is computed and finished. The study in [11] uses an updated version of the parallelization of the PSO method based on the distribution of the objective function evaluation and the available identified cores, which is based on the previous perspective. This viewpoint is used in this study.

3 Proposed Algorithm

The capacity of particles in the swarm to cover the search space and give consistent results will be restricted if the social factor is smaller than the cognitive element utilized in classic PSO algorithms (such as those demonstrated in [5]). As seen in [7], an incorrectly set inertial weight slows down the PSO algorithm's convergence speed. Furthermore, as the optimization issue becomes more intricate in its intricacies, the swarm performance degrades, resulting in a longer PSO method execution time.

By introducing the technique that will be proven in this work, the aforementioned issues are alleviated. The Parallelized linear TVAC and inertial weight of PSO, or "PLTVACIW-PSO," is a suggested method that takes use of the benefits of linearity for each time-variant acceleration coefficient described in [8,11] and the linearity of the inertial weight presented in [9]. PLTVACIW-PSO also employs the parallelization viewpoint described in [11,22]. Algorithm 1 shows the proposed PLTVACIW-PSO.

Algorithm 1: The Parallelized Linear Time-Variant Acceleration Coefficients and Inertial Weight of PSO (PLTVACIW-PSO) algorithm

---

Inputs: Objective function F, a swarm of particles $P = \{P_1, P_2, P_3, \ldots, P_n\}$, a number of processing unit "cores" $Co = \{Co_1, Co_2, Co_3, \ldots, Co_n\}$, predefined iterations $MAXITER$, iteration counter $iter = 1$.
Output: Optimized fitness value for a given function F.
Procedures:
1. The velocity and position V_P and X_P are initialized for all particles in the swarm.
2. The personal best $pbest_p$ and local best $gbest_p$ are determined for all particles in the swarm.
3. Repeat
   3.1- The objective function F evaluation is distributed across Cos.
   3.2 The global synchronization is done to ensure that the fitness of all particles in P is evaluated in a consistent manner.
   3.3 Personal best and global best are updated for all particles in the swarm.
   3.4 if ($iter \leq MAXITER$) then
   - Update c_1, c_2, and w using Eqs. (3)–(5) .
   - Updated c_1, c_2, and w will be used to update V_p and X_p using Eqs. (1) and (2), respectively.
   - $iter = iter + 1$
   - go to steps 3.1, 3.2, and 3.3.

   else
   - Final evaluated value for F is produced.

   end if
   Until ($MAXITER$ is reached or any stopping criterion)

The proposed method PLTVACIW-PSO takes a benchmark function as input and treats it as the goal function. The swarm iterates in PLTVACIW-PSO based on changes made to the inertial weight w and TVAC parameters, as well as parallelization of the objective function evaluation and the optimized value produced for the given objective function at each iteration; finally, the smallest optimized value is chosen from all values produced by all executed iterations.

4 Results, Analysis, and Discussions

There are two stages to the experiment described in this study. Phase one compares three kinds of algorithms to the proposed method PLTVACIW-PSO, which is utilized to optimize twelve distinct benchmark functions. The "IW family" is the first category, which contains algorithms that use PSO's parallelized and non-parallelized inertial weight. The "TVAC family," which comprises algorithms based on PSO's parallelized and non-parallelized TVAC, is the second category. Traditional algorithms such as PSO, differential evolution (DE), the genetic algorithm (GA), and the flower pollination algorithm (FPA) are included in the third category [11]. In phase two, the proposed PLTVACIW-PSO algorithm is compared against a group of algorithms that comprises two bio-inspired optimizations, the BAT and Multi-swarm BAT algorithms [16]. The experimental results found in [11,16] are also compared with the results achieved from the proposed algorithm. These functions' definitions and attributes are used to show how efficient and successful any particular optimization strategy [23,24]. The benchmark functions utilized in this study were chosen based on two criteria. Modality (single mode or multi-mode) and separability are two of these requirements (separable or non-separable). The multi-mode function has two or more local optima, while the unimodal function has just one. A separable function is one that has no interrelations between the variables that make up the function, while a non-separable function contains interrelations between the variables that make up the function. Twelve benchmark functions from [11,16] are utilized in this study to compare the proposed PLTVACIW-PSO method to the other algorithms listed in Tab. 1 (functions F11 and F12 are mentioned in [11] only).

4.1 Model Evaluation

The proposed PLTVACIW-PSO algorithm used in phase one is a stochastic iterative PSO algorithm by nature, which indicates that an optimized result will be produced during each iteration for any given function listed in Appendix A. Therefore, the values of Worst, Best, and Mean of Mean Square Error (MSE), represented in Eq. (6), are recorded for any set of iterations produced by any given function.

$$MSE = \frac{\sum_{i=1}^{n} (E_i - P_i)^2}{n} \tag{6}$$

where E_i is the actual experimental values, Pi are predictive values, and n are test data being used. The value of "Worst" indicates the most significant value of MSE along with the optimization iterations. The value of "Best" is the smallest MSE value given along with the optimization iterations. Finally, the value of "Mean" is delivered by calculating the average of all MSE values along with their optimization iterations.

A series of independent runs is used to generate accurate findings from the stochastic based methods indicated in Tab. 1 [11,16]. Fifty different algorithm executions are done in this work, and the best run is picked. This run provides the best overall results for the proposed PLTVACIW-PSO method and all other algorithms for Worst, Bestm, and Mean. In this experiment, the same fifty separate

runs were performed, and the best overall performance was chosen. Furthermore, for all algorithms, including the proposed PLIWTVAC-PSO, the average of the Worst, Best, and Mean values for the fifty separate executions is chosen for functions F1–3.

For the proposed PLIWTVAC-PSO, 100 separate executions are done, and the average values of Worst, Best, and Mean, as well as the best overall execution for the proposed PLIWTVAC-PSO, are chosen and compared to the BAT and MBAT algorithms.

4.2 Experimental Platform

In this paper, the implementation of both PLTVACIW-PSO algorithms was done using R language version 3.3.0, executed on a virtualized CentOS Linux operating system with a 2 GHz dual-core processor, 5 GB RAM, and 20 GB of non-volatile storage.

4.3 Experimental Parameters

For the IW family algorithms used in phase 1, inertial weight wrang = (0.4, 09), swarm size = 50, and iterations = 100. For the TVAC family algorithms used in phase one, inertial weight w = 0.721, c1rang = (1.28: 1.05), c2rang = (1.05: 1.28), and iterations = 100. For PSO, swarm size = 50, w = 0.721, c1 = 1.193, c2 = 1.193, and iterations = 100. For the DE algorithm used in phase I, population size = 50, crossover probability = 0.5, differential weighting factor = 0.8, and iterations = 100. For the GA used in phase I, population size = 50, crossover probability = 0.8, mutation probability = 0.1, and iterations = 100. For the FPA used in phase 1, population size = 25, probability switch = 0.8, and iterations = 100.

In phase II, the same configurations for phase I are used for PLTVACIW-PSO.

4.4 Result Analysis

Phase I

Three different experiments were conducted in phase one to illustrate the capabilities of the proposed PLTVACIW-PSO algorithm. The first experiment compares the proposed PLIWTVAC-PSO with the IW family mentioned earlier in Tab. 1. These experimental results are illustrated in Tab. 2.

Table 2: Performance of PLIWTVAC-PSO *vs.* the IW family (best execution of 50 independent executions)

Functions	Values	PLTVACIW-PSO	PLIW-PSO	LIW-PSO	PNLIW-PSO	NLIW-PSO
F1	Worst	**18.67**	18.94	18.72	19.18	18.9
	Best	**2.98**	3.98	3.75	3.65	4.59
	Mean	**6.5**	7.28	7.58	7.55	8.43
F2	Worst	**180.4**	266.8	250.5	237	265.02
	Best	**1.47**	2.11	1.95	2.31	2.41
	Mean	**16.46**	27.4	23.9	27.2	30.3
F3	Worst	**8.17E03**	1.15E04	9.90E03	1.26E04	8.78E03
	Best	**1.00E0**	1.94E0	1.85E0	6.65E0	2.96E0
	Mean	**4.38E02**	5.74E02	5.40E02	4.69E03	4.67E02

(Continued)

Table 2: Continued

Functions	Values	PLTVACIW-PSO	PLIW-PSO	LIW-PSO	PNLIW-PSO	NLIW-PSO
F4	Worst	**305.2**	330.80	331.5	331.5	324.7
	Best	**26.14**	105.4	100.2	59.7	83.1
	Mean	**127.02**	169	173.28	153.88	169.62
F5	Worst	**3.51E03**	1.15E05	1.79E05	6.15E04	1.55E05
	Best	1.25E − 01	**1.10E − 05**	1.26E − 04	1.84E − 04	6.28E − 04
	Mean	**1.65E02**	4.37E03	4.66E03	3.88E03	4.42E03
F6	Worst	**48.43**	135.6	114.7	152.9	106.5
	Best	0.09	0.04	0.04	**0.02**	0.03
	Mean	**1.80**	2.45	2.48	6.39	2.03
F7	Worst	**81.26**	98.71	96.61	112.2	94.99
	Best	**0.38**	1.28	0.55	0.61	0.78
	Mean	**6.26**	10.1	6.32	7.74	7.64
F8	Worst	**42.75**	43.43	44.82	61.91	46.9
	Best	**0.16**	0.54	0.60	2.02	0.78
	Mean	**6.4**	7.34	7.52	15.46	8.12
F9	Worst	**5.54E04**	6.03E04	6.41E04	6.08E04	6.93E04
	Best	**6.05E − 01**	2.47E0	2.09E0	9.28E − 01	2.12E0
	Mean	**2.72E03**	3.10E03	2.45E03	2.75E03	3.76E03
F10	Worst	**1.58**	1.64	1.63	1.62	1.61
	Best	**0.17**	0.33	0.51	0.77	0.52
	Mean	**0.53**	0.82	0.76	0.98	0.8
F11	Worst	**2.27E04**	2.68E04	2.49E04	2.81E04	2.46E04
	Best	**1.51E01**	7.35E01	7.40E01	1.18E02	2.13E01
	Mean	**2.07E03**	2.41E03	2.41E03	2.55E03	2.09E03
F12	Worst	**3.11E07**	4.78E07	4.12E07	4.47E07	4.95E07
	Best	**4.75E02**	4.10E02	4.09E02	1.60E03	2.01E03
	Mean	**1.42E06**	2.06E06	1.85E06	1.76E03	2.10E06

The second experiment compares the proposed PLIWTVAC-PSO with the TVAC family, including the PLTV-PSO algorithm; this family is mentioned in Tab. 1. These experimental results are illustrated in Tab. 3.

Table 3: Performance of PLIWTVAC-PSO *vs.* the TVAC family (best execution of 50 independent executions)

Functions	Values	PLTVACIW-PSO	PLTV-PSO	LTV-PSO	PNLTV-PSO	NLTV-PSO
F1	Worst	**18.67**	20.67	20.73	20.73	20.87
	Best	2.98	2.71	3.91	3.82	4.31
	Mean	**6.5**	9.02	9.8	9.97	10.09
F2	Worst	**180.4**	587.16	706.1	671.8	653.9
	Best	**1.47**	1.51	2.1	1.99	2.09
	Mean	**16.46**	65.36	65.36	62.08	72.79
F3	Worst	**8.17E03**	6.82E04	9.27E04	7.82E04	7.28E04
	Best	**1.00E0**	2.86E00	1.74E00	3.35E00	3.62E00
	Mean	**4.38E02**	2.75E03	3.79E03	2.84E03	3.32E03
F4	Worst	**305.2**	460.9	461.7	463.9	466.7
	Best	**26.14**	54.65	146.15	112.41	93.36
	Mean	**127.02**	189.59	227.32	208.47	205.96
F5	Worst	**3.51E03**	5.58E05	6.36E05	1.30E07	1.30E06
	Best	**1.25E − 01**	3.64E − 01	8.36E − 05	8.96E − 04	3.82E − 04
	Mean	**1.65E02**	2.30E04	2.80E04	1.94E05	2.88E04
F6	Worst	**48.43**	126.6	183.1	156.4	192.6
	Best	0.09	**0.02**	0.03	0.09	0.09
	Mean	**1.80**	6.39	7.79	7.21	7.43
F7	Worst	**81.26**	210.8	219.1	242.7	240.3
	Best	**0.38**	0.79	1.54	1.23	1.21
	Mean	**6.26**	20.85	22.96	23.78	24.94
F8	Worst	**42.75**	61.91	72.49	67.4	67.72
	Best	**0.16**	2.02	3.28	3.57	3.77
	Mean	**6.4**	15.46	18.12	17.17	18.94
F9	Worst	**5.54E04**	5.57E05	5.16E05	5.59E05	1.30E06
	Best	6.05E − 01	8.13E − 01	1.65E00	4.54E00	3.82E − 04
	Mean	**2.72E03**	2.32E04	2.37E04	2.49E04	2.88E04
F10	Worst	**1.58**	1.59	1.73	1.65	1.7
	Best	**0.17**	0.5	1	1.01	1.01
\	Mean	**0.53**	0.97	1.18	1.24	1.24
F11	Worst	**2.27E04**	6.39E04	8.39E04	8.12E04	7.83E04
	Best	**1.51E01**	5.17E01	8.37E01	8.33E01	7.17E01
	Mean	**2.07E03**	6.07E03	6.91E03	7.01E03	6.41E03

(Continued)

Table 3: Continued

Functions	Values	PLTVACIW-PSO	PLTV-PSO	LTV-PSO	PNLTV-PSO	NLTV-PSO
F12	Worst	**3.11E07**	2.61E08	3.06E08	3.69E08	3.11E08
	Best	**4.75E02**	1.29E03	2.22E03	1.64E03	2.28E03
	Mean	**1.42E06**	1.23E07	1.32E07	1.34E07	1.32E07

Finally, the third experiment aims at comparing the proposed PLIWTVAC-PSO with the traditional algorithms and the BAT algorithms as mentioned earlier in Tab. 1. The third experiment's results are presented in Tab. 4. The same 50 independent executions were also carried out in this experiment, and the best overall execution is determined.

Table 4: Performance of PLIWTVAC-PSO *vs.* traditional algorithms and the BAT algorithms (best execution of 50 independent executions)

Functions	Values	PLTVACIW-PSO	PSO	GA	DE	FPA	BAT	MBAT
F1	Worst	**18.67**	20.78	20.68	21.89	20.81	19.95	19.95
	Best	**2.98**	4.13	12.42	21.45	18.42	17.04	11.91
	Mean	**6.5**	10.46	17.02	21.69	19.87	19.43	15.79
F2	Worst	**180.4**	780.7	587.16	1695.23	651.64	317.48	324.29
	Best	**1.47**	2.95	29.05	1269.96	104.78	120.66	30.54
	Mean	**16.46**	79.97	159.34	1466.61	297.13	257.51	91.67
F3	Worst	**8.17E03**	9.99E + 04	7.38E + 04	4.85E + 05	7.86E + 04	3.40E + 05	4.20E + 04
	Best	**1.00E0**	5.37E + 00	1.87E + 02	2.74E + 05	1.78E + 03	5.20E + 03	3.00E + 03
	Mean	**4.38E02**	3.41E + 03	1.05E + 04	3.87E + 05	1.83E + 04	3.70E + 05	4.10E + 04
F4	Worst	305.2	497.4	474.25	888.75	436.37	**303.88**	**303.88**
	Best	**26.14**	91.23	154.06	686.73	265.6	179.98	**51.11**
	Mean	127.02	211.18	245.58	779.01	347.81	252.81	**93.2**
F5	Worst	**3.51E03**	1.28E + 06	7.92E + 05	1.80E + 10	6.16E + 06	**4.00E + 03**	3.40E + 05
	Best	**1.25E − 01**	7.34E + 04	1.50E − 02	1.64E + 10	2.10E + 01	843.7	77.47
	Mean	**1.65E02**	4.13E + 04	3.28E + 04	1.35E + 10	1.25E + 05	6.10E + 05	**847.42**
F6	Worst	**48.43**	180.2	149.72	648.07	229.28	85.31	85.31
	Best	0.09	**0.07**	1.41	417.57	5.53	2.03	0.24
	Mean	**1.80**	7.77	23.55	531.11	50.23	26.52	3.46
F7	Worst	**81.26**	221.2	234.51	970.31	240.11	85.31	4.20E + 04
	Best	**0.38**	1.76	16.42	689.64	67.67	40.31	6.86
	Mean	**6.26**	26.07	62.92	852.97	122.61	93.88	25.02
F8	Worst	42.75	68.01	65.79	162.11	69.15	31.88	**19.95**
	Best	**0.16**	4.13	16.16	65.76	34.41	25.67	6.07
	Mean	**6.4**	17.05	31.08	125.18	46.94	36.49	14.31
F9	Worst	5.54E + 04	7.54E + 05	3.73E + 05	2.44E + 06	5.36E + 05	3.40E + 05	3.40E + 05
	Best	**6.05E − 01**	3.09E + 00	2.27E + 03	1.60E + 06	1.76E + 04	1.50E + 04	1.00E + 03
	Mean	**2.72E03**	2.76E + 04	7.70E + 04	1.99E + 06	1.55E + 05	1.00E + 05	1.20E + 04
F10	Worst	1.58	1.74	**1.53**	3.87	1.62	1.10E + 06	1.10E + 06
	Best	**0.17**	1.02	0.75	2.46	0.01	1.09	0.54
	Mean	**0.53**	1.29	1.02	3.47	0.53	3.69	3.09

(Continued)

Table 4: Continued

Functions	Values	PLTVACIW -PSO	PSO	GA	DE	FPA	BAT	MBAT
F11	Worst	**2.27E04**	8.89E + 04	6.97E + 04	1.75E + 05	6.73E + 04		
	Best	**1.51E01**	1.69E + 02	4.03E + 04	9.79E + 04	1.03E + 04		
	Mean	**2.07E03**	8.24E + 03	2.09E + 04	1.38E + 05	3.21E + 04		
F12	Worst	**3.11E07**	4.05E + 08	2.71E + 08	1.17E + 09	2.88E + 08		
	Best	**4.75E02**	2.14E + 03	1.88E + 06	7.41E + 08	1.04E + 07		
	Mean	**1.42E06**	1.42E + 07	4.83E + 07	9.36E + 08	8.31E + 07		

Setting the IW to the highest value, which permits particles to travel at a fast speed and participate in extensive exploration, increases the exploration behavior of particles in the proposed PLIWTVAC-PSO algorithm towards the nearest position of the optimal value. Following that, IW repeatedly declines linearly to lower levels, causing the particles to travel slowly and engage in more deliberate investigation. As a consequence, lowering the Worst values acquired in early iterations provides the swarm with greater Best values during its final rounds than other IW family algorithms. Furthermore, the mean value indicates that the proposed PLIWTVAC-PSO yields modest Worst values in early iterations and converges to optimal Best values in middle and final iterations. Tab. 3 demonstrates that there are many more improvements to the PLTV-PSO algorithm by linearly decreasing the IW method in the form of the proposed PLIWTVAC-PSO. The linear TVAC and IW may help the swarm's overall behavior. When linear TVAC is combined with linearly decreasing IW, the exploration behavior of all particles in the swarm is significantly improved; the worst values obtained during earlier iterations are significantly reduced, and the best values are obtained during the final iterations, affecting the overall Mean values obtained by all iterations of the proposed PLIWTVAC-PSO. Tab. 4 shows demonstrates the suggested PLIWTVAC-PSO outperforms other standard algorithms for the Worst, Best, and Mean values, respectively, than PSO, DE, GA, and FPA algorithms. Other algorithms, like as GA and DE, have a lot more parameters to tune, such as crossover and mutation probabilities.

The average values of the Worst, Best, and Mean are acquired from 50 different executions for the proposed PLIWTVAC-PSO, IW family algorithms, TVAC family algorithms, and conventional algorithms to verify that the proposed algorithm PLIWTVAC-PSO obtains trustworthy results. Functions F1–3 are used in this supplementary experiment. These findings are shown in Tabs. 5–7. The data in Tabs. 5–7 demonstrate that the proposed PLIWTVAC-PSO algorithm is successful, and that the linearity of both TVAC and IW combined effectively manages the swarm toward the best outcomes. Furthermore, the optimized outcomes from 50 separate executions do not differ much from one to the next.

Table 5: Performance of PLIWTVAC-PSO *vs.* the IW family (average of worst, best, and mean values of 50 independent executions)

Functions	Values	PLTVACIW-PSO	PLIW-PSO	LIW-PSO	PNLIW-PSO	NLIW-PSO
F1	Worst	**18.90**	20.56	20.77	20.86	20.92
	Best	**3.81**	4.71	4.64	4.62	4.76
	Mean	**7.75**	8.96	8.90	8.91	9.88

(Continued)

Table 5: Continued

Functions	Values	PLTVACIW-PSO	PLIW-PSO	LIW-PSO	PNLIW-PSO	NLIW-PSO
F2	Worst	**231.35**	232.56	238.92	232.27	232.65
	Best	**1.64**	1.68	1.82	1.73	1.81
	Mean	**21.19**	21.61	22.67	21.26	21.55
F3	Worst	**1.06E + 04**	1.06E + 04	1.06E + 04	1.06E + 04	1.07E + 04
	Best	**2.15E + 00**	2.56E + 00	2.41E + 00	2.43E + 00	2.71E + 00
	Mean	**4.39E + 02**	4.66E + 02	4.51E + 02	4.43E + 02	4.46E + 02

Table 6: Performance of PLIWTVAC-PSO *vs.* the TVAC family (average of worst, best, and mean values of 50 independent executions)

Functions	Values	PLTVACIW-PSO	PLTV-PSO	LTV-PSO	PNLTV-PSO	NLTV-PSO
F1	Worst	**18.90**	21.76	21.9	21.84	21.82
	Best	**3.81**	3.62	4.81	4.76	5.77
	Mean	**7.75**	11.46	11.77	1.85	12.23
F2	Worst	**231.35**	675.81	886.54	891.97	865.28
	Best	**1.64**	1.64	1.63	1.62	1.55
	Mean	**21.19**	73.56	83.56	82.51	82.58
F3	Worst	**1.06E + 04**	7.90E + 04	8.03E + 04	8.31E + 04	8.22E + 04
	Best	**2.15E + 00**	5.77E + 00	6.41E + 00	4.55E + 00	4.04E + 00
	Mean	**4.39E + 02**	4.20E + 03	4.27E + 03	4.41E + 03	4.34E + 03

Table 7: Performance of PLIWTVAC-PSO *vs.* traditional algorithms (average of the worst, best, and mean values of 50 independent executions)

Functions	Values	PLTVACIW-PSO	PSO	GA	DE	FPA
F1	Worst	**18.90**	22.78	22.74	24.80	22.82
	Best	**3.81**	5.31	14.03	22.45	19.42
	Mean	**7.75**	12.52	19.98	23.69	20.39
F2	Worst	**231.35**	871.85	2720.27	690.13	770.07
	Best	**1.64**	1.68	1750.84	40.23	80.53
	Mean	**21.19**	74.78	1907.92	90.71	357.11
F3	Worst	**1.06E + 04**	1.09E + 04	8.89E + 05	6.80E + 06	1.51E + 05
	Best	**2.15E + 00**	7.98E + 00	2.35E + 02	2.11E + 05	4.77E + 03
	Mean	**4.39E + 02**	4.89E + 03	3.76E + 04	8.91E + 05	5.91E + 04

Phase II

We employed ten benchmark medical datasets in this phase. The datasets were taken from the University of California, Irvine's learning machine server [25]. A short explanation of each chosen dataset may be found in Tab. 8. As can be seen, several entries in the adopted data sets are missing. In this article, the median value of all known values of the functional class is substituted by all of these missing values. The mathematical definition of the median is as follows:

$$\overline{s}_{i,j} = median_{i:s_{i,j} \in W_r s_{i,j}} \tag{7}$$

where si, j is the missing value for a specific ith class W's jth feature. For missing categorical values, the most typically replaced value is the missing value for a specific class function.

Table 8: Medical dataset description

ID	Dataset	No. of instances	No. of features	Total no. of classes
DS1	Erythemato-squamous	366	34	6
DS2	Breast cancer	569	32	2
DS3	Hepatitis	155	19	2
DS4	SPECTF heart data set	267	45	2
DS5	Cervical cancer (Risk factors)	858	36	2
DS6	Parkinson	1208	26	2
DS7	Lung cancer	32	56	3
DS8	Thyroid disease	7200	21	3
DS9	Hepatitis C Virus (HCV) for Egyptian patients	1385	29	4
DS10	Parkinson's disease classification	197	23	2

Three parameters are used to test PLIWTVAC-PSO selected features. Other requirements include accuracy of classification, number of specified features and CPU processing time have been considered [26]. Tab. 9 provides a distinction between feature selection, classification accuracy and processing time before and after applying PLIWTVAC-PSO.

Table 9: Feature selection and classification accuracy before and after applying PLIWTVAC-PSO for medical datasets

	Before applying PLIWTVAC-PSO			After applying PLIWTVAC-PSO		
Datasets	Total no. of features	Accuracy (%)	Time (sec)	Selected features	Accuracy (%)	Time (sec)
DS1	34	91.9814	0.7199	21	94.9467	0.4129
DS2	32	90.2838	0.0969	14	94.6010	0.0477
DS3	19	83.7006	0.0939	5	91.0359	0.0356

(Continued)

Table 9: Continued

Datasets	Before applying PLIWTVAC-PSO			After applying PLIWTVAC-PSO		
	Total no. of features	Accuracy (%)	Time (sec)	Selected features	Accuracy (%)	Time (sec)
DS4	45	75.8929	0.0992	20	81.5087	0.0371
DS5	36	63.9529	0.0995	18	74.3023	0.0523
DS6	26	89.5345	0.2912	15	92.3475	0.1986
DS7	56	87.2746	0.0950	20	90.7859	0.0355
DS8	21	89.5345	0.2912	13	93.6982	0.1986
DS9	29	88.1434	0.1230	14	91.1285	0.1023
DS10	23	85.1255	0.5118	10	92.9413	0.4269

The number of features and the mean accuracy of 10 folds are presented together with CPU processing time. Only a subset of characteristics may reduce classification efficiency, but computation time and storage can be greatly reduced. Fig. 1 depicts a comparison of feature selection and classification accuracy before and after PLIWTVAC-PSO application. Such findings illustrate the superiority of the PLIWTVAC-PSO selection method. These traits may also be utilized to increase clustering performance. Clusters made consisting of a selection of important qualities are more realistic and understandable than clusters made up of all of them, such as noise. It may also aid in the interpretation and comprehension of facts.

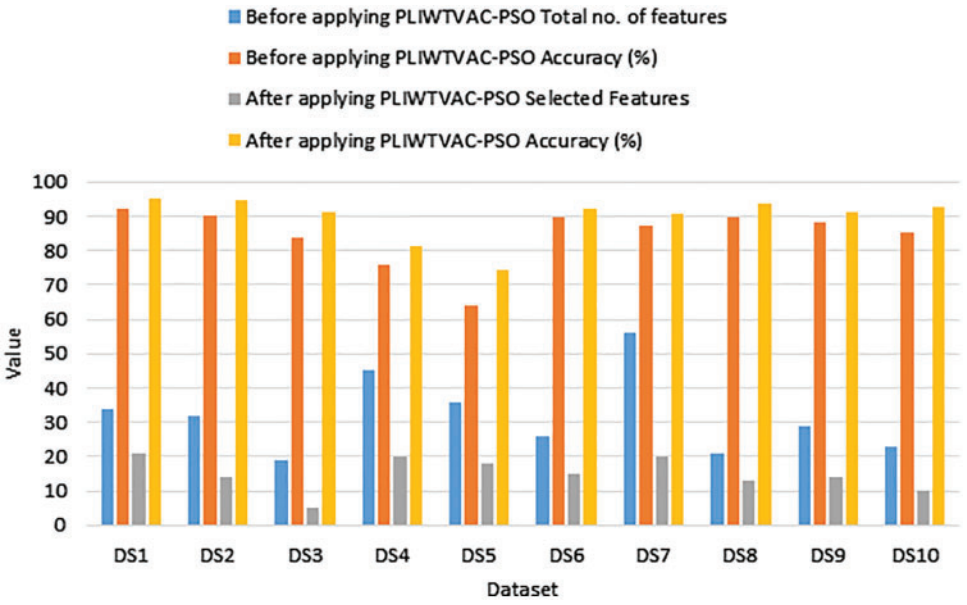

Figure 1: Feature selection and classification accuracy before and after applying PLIWTVAC-PSO for medical datasets

5 Conclusions

The suggested PLTVACIW-PSO algorithm outperforms PSO IW-based algorithms, TVAC algorithms, GA and DE algorithms, and even multi-swarm-based algorithms in terms of efficiency and efficacy. This demonstrates that combining the linearity of IW and TVAC may adjust a single swarm-based PSO algorithm to get optimal outcomes. Furthermore, the suggested PLTVACIW-PSO method tends to reduce the worst values from earlier iterations in order to attain the best value during the final rounds, demonstrating that the proposed algorithm is an effective and reliable global optimization technique. The suggested PLTVACIW-PSO Algorithm was used to verify ten medical datasets. The findings suggest that PLTVACIW-PSO can increase classification, consistency, the number of features picked, and convergence speed by a substantial amount.

Future work could include switching between the linearity of IW and the non-linearity of TVAC and vice versa, replacing PSO parallelization across multiple cores with a dedicated board of CPUs, and finally, instead of using a single swarm, the proposed PLTVACIW-PSO can use multiple swarms of particles. PLTVACIW-results PSO's may be used to verify increasingly difficult challenges in science and contemporary engineering in the future.

Acknowledgement: The authors would like to thank Prince Sultan University, Riyadh, Saudi Arabia for paying the Article Processing Charges (APC) of this publication. Special acknowledgement to Automated Systems & Soft Computing Lab (ASSCL), Prince Sultan University, Riyadh, Saudi Arabia. Also, the authors wish to acknowledge the editor and anonymous reviewers for their insightful comments, which have improved the quality of this publication.

Funding Statement: This research was funded by the Prince Sultan University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] C. Eberhart, Y. Shi and J. Kennedy, *Swarm Intelligence*, Burlington, MA: Morgan Kaufmann, 2001.
[2] G. Beni and J. Wang, *Swarm Intelligence*, Tokyo, Japan: RSJ Press, pp. 425–428, 1989.
[3] M. Ab-Wahab, S. Nefti-Meziani and A. Atyabi, "A comprehensive review of swarm optimization algorithms," *PLoS One*, vol. 10, no. 5, pp. e0122827, 2015.
[4] C. Blum and D. Merkle, *Swarm Intelligence: Introduction and Applications*, Heidelberg, Germany: Springer verlag, 2008.
[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Internal Conf. on Neural Networks*, Perth Australia, vol. 4., pp. 1942–1948, 1995.
[6] S. Rana, S. Jasola and R. Kumar, "A review on particle swarm optimization algorithms and their applications to data clustering," *Artificial Intelligence Revision*, vol. 35, pp. 211–222, 2011.
[7] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *IEEE Int. Conf. on Evolutionary Computation Proc., IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, Anchorage, AK, USA, USA, 4–9 May 1998. https://doi.org/10.1109/ICEC.1998.699146.
[8] A. Ratnaweera, S. K. Halgamuge and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.
[9] Y. Zheng, L. Ma, L. Zhang and J. Qian, "Empirical study of particle swarm optimizer with an increasing inertia weight," in *2003 Congress on Evolutionary Computation (CEC '03)*, Canberra, ACT, Australia, 8–12 Dec. 2003. https://doi.org/10.1109/CEC.2003.1299578.

[10] E. -S. M. El-kenawy and M. Eid, "Hybrid gray wolf and particle swarm optimization for feature selection," *International Journal of Innovative Computing, Information and Control*, vol. 16, no. 3, pp. 831–844, 2020.
[11] K. Fouad, T. Elsheshtawy and M. Dawood, "Parallelized linear time-variant acceleration coefficients of PSO algorithm for global optimization," in *12th Int. Conf. on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, 21–22 Oct. 2016. https://doi.org/10.1109/ICCES.2017.8275369.
[12] P. Chen, C. Wu, Y. Fu and C. KO, "A PSO method with nonlinear time-varying evolution for the optimal design of PID controllers in a pendubot system," *Artif Life Robotics*, vol. 14, no. 1, pp. 58–61, 2009.
[13] K. Price, R. Storn and J. Lampinen, *Differential Evolution-A Practical Approach to Global Optimization*, Berlin Heidelberg: Springer-Verlag, ISBN 3540209506, 2006.
[14] E. -S. M. El-kenawy, A. Ibrahim, S. Mirjalili, Y. Zhang, S. Elnazer *et al.,* "Optimized ensemble algorithm for predicting metamaterial antenna parameters," *Computers Materials & Continua*, vol. 71, pp. 4989–5003, 2022.
[15] X. S. Yang, "Flower pollination algorithm for global optimization," in J. Durand-Lose and N. Jonoska (Eds.) *Unconventional Computation and Natural Computation. UCNC 2012*, Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, vol. 7445, pp. 240–249, 2012. https://doi.org/10.1007/978-3-642-32894-7_27.
[16] G. Wang, B. Chang and Z. Zhang, "A multi-swarm bat algorithm for global optimization," in *IEEE Congress on Evolutionary Computation (CEC 2015)*, Sendai, Japan, 25–28 May 2015. https://doi.org/10.1109/CEC.2015.7256928.
[17] H. Ran, W. Yong-Ji, W. Qing, Z. Jin-Hui and H. Chen-Yong, "An improved particle swarm optimization based on self-adaptive escape velocity," *Journal of Software*, vol. 16, no. 12, pp. 2036–2044, 2005.
[18] C. Ke, Z. Fengyu, Y. Lei, W. Shuqian and W. Yugang, "A hybrid particle swarm optimizer with sine cosine acceleration coefficients," *Information Sciences*, vol. 422, pp. 218–241, 2018.
[19] Z. Tang and D. Zhang, "A modified particle swarm optimization with adaptive acceleration coefficients," in *Proc. of the Asia-Pacific Conf. on Information Processing (APCIP'09)*, Shenzhen, China, pp. 330–332, 18–19 July 2009.
[20] A. Ibrahim, H. A. Ali, M. M. Eid and E. -S. M. El-Kenawy, "Chaotic harris hawks optimization for unconstrained function optimization," in *2020 16th Int. Computer Engineering Conf. (ICENCO)*, Cairo, Egypt, IEEE, pp. 153–158, 2020.
[21] B. Qinghai, "Analysis of particle swarm optimization algorithm," *Computer and Information Science*, vol. 3, no. 1, pp. 180–184, 2010.
[22] J. Schutte, J. Reinbolt, B. Fregly, R. Haftka and A. George, "Parallel global optimization with particle swarm algorithm," *Int. J. Numerical Methods in Engineering*, vol. 61, pp. 2296–231, 2004.
[23] M. Jamil, X. Yang and H. Zepernick, "Test functions for global optimization: Comprehensive survey," in X, Yang, Z. Cui, R. Xiao, A. Gandomi and M. Karamanoglu (Eds.) *Swarm Intelligence and Bio-Inspired Computation*, Waltham, MA: Elsevier, pp. 193–222, 2013. https://doi.org/10.1016/B978-0-12-405163-8.00008-9.
[24] X. Yao, Y. Liu and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.
[25] D. Dua and C. Graff, *UCI Machine Learning Repository*, Irvine, CA: University of California, School of Information and Computer Science, 2019. http://archive.ics.uci.edu/ml.
[26] H. Sun and R. Grishman, "Lexicalized dependency paths based supervised learning for relation extraction," *Computer Systems Science and Engineering*, vol. 43, no. 3, pp. 861–870, 2022.