Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 560614, 13 pages
http://dx.doi.org/10.1155/2013/560614

Research Article

CFSO^3: A New Supervised Swarm-Based Optimization Algorithm

Antonino Laudani, Francesco Riganti Fulginei, Alessandro Salvini, Maurizio Schmid, and Silvia Conforto

Department of Engineering, Roma Tre University, Via V. Volterra 62, 00146 Rome, Italy

Correspondence should be addressed to Antonino Laudani; alaudani@uniroma3.it

Received 3 May 2013; Accepted 7 July 2013

Academic Editor: Orwa Jaber Housheya

We present CFSO^3, an optimization heuristic within the class of the swarm intelligence, based on a synergy among three different features of the Continuous Flock-of-Starlings Optimization. One of the main novelties is that this optimizer is no more a classical numerical algorithm since it now can be seen as a continuous dynamic system, which can be treated by using all the mathematical instruments available for managing state equations. In addition, CFSO^3 allows passing from stochastic approaches to supervised deterministic ones since the random updating of parameters, a typical feature for numerical swam-based optimization algorithms, is now fully substituted by a supervised strategy: in CFSO^3 the tuning of parameters is *a priori* designed for obtaining both exploration and exploitation. Indeed the exploration, that is, the escaping from a local minimum, as well as the convergence and the refinement to a solution can be designed simply by managing the eigenvalues of the CFSO state equations. Virtually in CFSO^3, just the initial values of positions and velocities of the swarm members have to be randomly assigned. Both standard and parallel versions of CFSO^3 together with validations on classical benchmarks are presented.

1. Introduction

Exploration and exploitation are the two fundamental requirements for the algorithms devoted to inverse problems and/or optimization. In many applications more than one global optimum could exist, and/or the space of the solutions to be investigated could show a very large dimension. In all these cases, the risk of remaining entrapped into a local minimum is quite high, especially if the algorithm in use is not able to explore large spaces [1]. On the other hand, in other cases, it is necessary to converge to a solution granting a high level of accuracy. These conditions are difficult to be satisfied if the algorithm is not able to perform good exploitation [2]. Thus, apparently, exploration and exploitation seem to be antithetic requirements. With the aim to match these two properties in only one strategy of optimization, we aim to present in this paper CFSO^3 that takes its main inspiration from swarm algorithms and in particular from their translation into continuous dynamic systems.

From a historical point of view, the swarm optimization has been introduced by Kennedy and Eberhart [3] in the 1990s with the Particle Swarm Optimization (PSO), a heuristic inspired to the social and collective behaviour shown by several animal species such as flock of birds or school of fishes [4, 5]. Virtually, it consists of a one-to-one correspondence between the motion of flocks searching for food and the iterative steps of algorithms searching for the best solution for optimization. Many authors have published a large quantity of works related to PSO, and the vastness of applications into different fields of science, such as engineering, physics, chemistry, artificial intelligence, and economics, testifies its success among many scientific communities (e.g., see [6–9] and the references within). A large series of changes from the original PSO have been proposed in order to improve its performances. In particular, many works focused on the way to manage the tuning of parameters for achieving better convergence to the global optimum and/or for improving exploration for multimodal problems (e.g., see [10–19] and the reference within), and the effects of topological rules on performances have been discussed [20–26]. The basic idea of topological rules is to link each member of the swarm with others by generating more complex information exchange among particles than the simple use of the global best. A particular way to link the particle behaviors is achieved

by using a kind of neighbor-velocity matching, that is, the idea of the Flock-of-Starlings Optimization (FSO) [27, 28], inspired by a naturalistic work presented in [29]. The FSO uses topology for exchanging the information about the current velocity of each connected particle/bird. Recently, many authors have proposed several attempts to treat the swarm numerical algorithms as continuous dynamical systems. For example, important contributions are in [30–32]. The main goal of these works is to investigate the stability of PSO and the effects on the optimization performance produced by different settings of the parameters. In this scenario, the authors of the present paper have translated the numerical FSO equations into the state equations of a kinetic dynamic system in the time domain (continuum), by proposing the so-called Continuous Flock-of-Starlings Optimization (CFSO) [33–35]. This new optimizer, that is, the starting point of the present paper, turns out to be quite effective, thanks to the use of mathematical closed forms which describe the swarm member trajectories as function of time, since it opens the road to all those mathematical instruments used for studying state equations. In addition, CFSO has several features that make it very interesting and promising such as, among other ones, the setting of parameters without using randomness. Thanks to the stability analysis of the CFSO state equations, the tuning of the parameters becomes the way for controlling the stability of the trajectories within the space of solutions, that is, for controlling the exploration and the exploitation, since the parameter values fix the values of the eigenvalues of the dynamic system (e.g., see [35–38] and the reference within).

In this paper, results derived from investigation on this distinguishing behaviour of the CFSO are discussed, in order to verify the effectiveness of the innovative nonstochastic approach characterizing this continuous swarm optimizer. In particular, the present paper shows a hybridization of three different features of CFSO, which allow us to enhance both exploration and convergence. Indeed, the changing from exalting exploration to giving a boost to the convergence/refinement is achievable simply by means of a suitable setting of the real part of pole values (or the eigenvalues) related to the CFSO state equations: in particular *pure imaginary poles* perform the exploration of the whole space of solutions; *instable poles* allow to escape from local minima, whereas refinement of the solution is obtained by launching an *asymptotically stable* CFSO. Only the initial position and velocity of each individual (birds) are randomly assigned.

Validations on classical benchmarks in order to show the advantages of the proposed continuous approach are presented, and the CFSO implementation has been made available for downloading at the link indicated in [39].

2. Recall of the Continuous Flock-of-Starlings Optimization: From Numerical Algorithms to Dynamic System State Equations

In [33, 35] it has been proved that it is possible to convert the numerical swarm optimization algorithm, PSO or FSO, into continuous time-domain dynamical kinetic systems described by a set of state equations. Virtually, the state equations equivalent to the rules used by the swarm algorithms for updating velocity and position of generic kth particle are

$$\dot{v}_k^j(t) = \omega v_k^j(t) + \lambda\left(p_{\text{best}_k}^j(t) - x_k^j(t)\right) + \gamma\left(g_{\text{best}}^j(t) - x_k^j(t)\right) + \sum_{m=1}^{N} h_{km} v_m^j(t), \tag{1}$$

$$\dot{x}_k^j(t) = v_k^j(t), \tag{2}$$

where $j = 1 \cdots \Delta$, Δ is the dimension of the solution space, and ω, λ, and γ are the so-called inertial, cognitive, and social coefficients, respectively; $g_{\text{best}}^j(t)$ is the jth component of the global best of the whole swarm, whereas $p_{\text{best}_k}^j(t)$ is the jth component of the personal best of the kth particle; $\sum_{m=1}^{N} h_{km} v_m^j(t)$ ($h_{km} = h$ if the kth bird is controlling the mth one, $h_{km} = 0$ otherwise) is the term that transforms the PSO into the FSO. This term, which strongly modifies the collective behaviour of the flock as shown in [27, 28], comes from the observation [29] that, in real flock each generic bird controls and follows the flight of a number $N_{\text{ctrl\_birds}}$ of other members of the flock, no matter what are their positions inside the flock. Finally, the personal best and global best have been assumed to be the excitations of the dynamic system by writing the quantity

$$\mathfrak{I}_k^j(t) = \lambda \cdot p_{\text{best}_k}^j(t) + \gamma \cdot g_{\text{best}}^j(t). \tag{3}$$

By posing $\mu = \lambda + \gamma$ it is possible to rewrite (1) in the following form:

$$\dot{v}_k^j(t) = \omega v_k^j(t) - \mu x_k^j(t) + \sum_{m=1}^{N} h_{km} v_m^j(t) + \mathfrak{I}_k(t). \tag{4}$$

Equations (2) and (4) describe the state equations of a continuous dynamic system. Thus, if we consider a flock made of N birds, the state-equation system of the CFSO has dimension $2N \times 2N$, and it can be expressed, for each component related to the jth dimension, as follows (from now on, in order to simplify the notation the apex j will be dropped; consequently, all expressions must be implicitly assumed to be valid for each jth generic component):

$$\begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\mathbf{x}} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \mathbf{1} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{x} \end{bmatrix} + \begin{bmatrix} \mathbf{F} \\ \mathbf{0} \end{bmatrix}. \tag{5}$$

Submatrix $\mathbf{1}$ appearing in (5) is the $N \times N$ identity matrix. $N \times N$ square submatrix $\mathbf{A}_{1,1}$ is defined as

$$\mathbf{A}_{1,1} = \omega \cdot 1 + \mathbf{H}. \tag{6}$$

Matrix $\mathbf{H}$ takes into account the neighbor-velocity-matching rule and for this reason is called neighbor-velocity-matching matrix. The nonzero values of $\mathbf{H}$ are the nondiagonal entries for which $h_{km} = h$ this means that the kth bird is controlling the velocity of the mth one. It is evident that, just considering the absence of the matrix $\mathbf{H}$ in (6),

it is possible to commutate the CFSO to a continuous PSO (CPSO). Finally, the last $N \times N$ submatrix appearing in (5) is

$$\mathbf{A}_{1,2} = -\mu \cdot \mathbf{1}, \tag{7}$$

whereas vector $\mathbf{F}$ has each kth row-entry $\Im_k(t)$ of (3).

2.1. Solution of the CFSO State Equations. Let us assume that the space where the solution lies is bounded, and then, also the global and personal best positions will be bounded and consequently we can apply the Laplace Transform, $\pounds\{\bullet\}$, in the variable $s \in \mathbb{C}$ to the system (5) as follows:

$$s\begin{bmatrix}\mathbf{V}(s)\\ \mathbf{X}(s)\end{bmatrix} = \begin{bmatrix}\mathbf{A}_{1,1} & \mathbf{A}_{1,2}\\ \mathbf{1} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{V}(s)\\ \mathbf{X}(s)\end{bmatrix} + \begin{bmatrix}\mathbf{F}(s)\\ \mathbf{0}\end{bmatrix} + \begin{bmatrix}\mathbf{v}(0)\\ \mathbf{x}(0)\end{bmatrix}. \tag{8}$$

In (8), both $\mathbf{v}(0)$ and $\mathbf{x}(0)$ are column vectors (from now on, for briefness, we will indicate $\mathbf{x} = \mathbf{x}(0)$ and $\mathbf{v} = \mathbf{v}(0)$). They take into account the initial conditions related to velocities and positions, respectively, while each single $\pounds\{\bullet\}$ is

$$\mathbf{V}(s) = \pounds\{\mathbf{v}(t)\} = \int_0^{+\infty} \mathbf{v}(t)\, e^{-st} dt, \tag{9}$$

$$\mathbf{X}(s) = \pounds\{\mathbf{x}(t)\} = \int_0^{+\infty} \mathbf{x}(t)\, e^{-st} dt, \tag{10}$$

$$\mathbf{F}(s) = \pounds\{\mathbf{F}(t)\} = \int_0^{+\infty} \mathbf{F}(t)\, e^{-st} dt. \tag{11}$$

Finally, by solving (8) in the Laplace domain for $\mathbf{X}(s)$, we have

$$\mathbf{X}(s) = \left(s^2 1 - \mathbf{A}_{1,1}s - \mathbf{A}_{1,2}\right)^{-1} \cdot \left\{(s1 - \mathbf{A}_{1,1}) \cdot \mathbf{x} + \mathbf{v} + \mathbf{F}\right\}. \tag{12}$$

Now, let us now decompose the response (solution) for the kth bird as the superposition of the free response, $\mathbf{X}^{\text{free}}(s)$, and the forced response, $\mathbf{X}^{\text{forced}}(s)$; that is, $\mathbf{X}(s) = \mathbf{X}^{\text{free}}(s) + \mathbf{X}^{\text{forced}}(s)$. Thus, we have

$$\mathbf{X}^{\text{free}}(s) = \left(s^2 1 - \mathbf{A}_{1,1}s - \mathbf{A}_{1,2}\right)^{-1} \cdot \left\{(s1 - \mathbf{A}_{1,1}) \cdot \mathbf{x} + \mathbf{v}\right\}, \tag{13}$$

$$\mathbf{X}^{\text{forced}}(s) = \left(s^2 1 - \mathbf{A}_{1,1}s - \mathbf{A}_{1,2}\right)^{-1} \cdot \mathbf{F}(s). \tag{14}$$

In this way, we can collect together global and all personal bests inside the forced solution (14), whereas the initial conditions appear just in the free solution (13). This is a crucial fact since it means that, for studying the collective behaviour of the swarm, we have to analyze just the free response (13) that is not influenced by the habitat (i.e., global and personal bests valuated by the fitness function), but just by the initial conditions related to each swarm member. Clearly, after having written the solution of the state equations in the Laplace domain, we have to translate the solution in the time domain. But the Laplace anti-transformation is not always possible, since the inverse of matrix $(s^2 1 - \mathbf{A}_{1,1}s - \mathbf{A}_{1,2})$ could not be simply evaluated in an analytical way. Indeed, if we write the neighbour velocity-matching, that is, matrix ($\mathbf{H}$), without a pre-fixed order, the only way to solve the CFSO equations in the time domain is to adopt a numeric ordinary differential equation solver, for example, by means of a numerical Runge-Kutta algorithm (ODE suite of MATLAB). On the other hand, it has been proved in [35] that a particular ordered choice of the matrix describing the neighbour-velocity matching allows to analytically integrate the CFSO equations, by means of the inverse Laplace transforming, and closed-form expressions, in the time domain, are easily available. The fundamental advantage using closed forms is that they remarkably reduce the computational cost making easier the setup of the optimization problem. In addition they can be directly used to update position and velocity, as it will be shown in next sections.

Let us show herein the way in which it is possible to achieve closed forms. For simplicity, but without loss of generality, let us consider the so-called *fully connected* CFSO, which occurs if $N_{\text{ctrl\_birds}} = N - 1$; that is, each kth bird controls all the other $N - 1$ members of the flock. This means that all the non-diagonal entries of the neighbour-velocity-matching matrix $\mathbf{H}$ are set to a value equal to h:

$$h_{k,m} = \begin{cases} h & k \neq m \\ 0 & k = m. \end{cases} \tag{15}$$

Thus, the matrices appearing in the solution of the system (12) become, respectively,

$$s^2 1 - \mathbf{A}_{1,1}s - \mathbf{A}_{1,2} = \begin{pmatrix} s^2 - \omega s + \mu & -hs & \cdots & -hs \\ -hs & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -hs \\ -hs & \cdots & -hs & s^2 - \omega s + \mu \end{pmatrix}, \tag{16}$$

$$s1 - \mathbf{A}_{1,1} = \begin{pmatrix} s - \omega & -h & \cdots & -h \\ -h & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -h \\ -h & \cdots & -h & s - \omega \end{pmatrix}. \tag{17}$$

The values of the entries of the inverse, $[C]$, related to the symmetric matrix (16) are

$$c_{k,m} = \begin{cases} \dfrac{s^2 - \omega s + \mu - (N-2)hs}{\left(s^2 - (\omega + (N-1)h)s + \mu\right)\left(s^2 + (h - \omega)s + \mu\right)} & \text{if } k = m \\ \dfrac{hs}{\left(s^2 - (\omega + (N-1)h)s + \mu\right)\left(s^2 + (h - \omega)s + \mu\right)} & \text{if } k \neq m. \end{cases} \tag{18}$$

Thus, by compacting in one single term

$$a_k = -(\omega - h)\, x_k - h \sum_{m=1}^{N} x_m + v_k, \tag{19}$$

the solution can be finally expressed by

$$\mathbf{X}(s) = [C] \cdot [s\mathbf{x} + \mathbf{a} + \mathbf{F}(s)]. \tag{20}$$

2.2. Time Windowing. Now, in order to solve the system of state equations (20), a further consideration must be done since the force term (vector **F**) is not *a priori* known: its elements depend on the trajectories followed by the birds. Consequently they must be evaluated dynamically according to the several fitness values which are monitored during the motion of the swarm/flock. Moreover, it is worth noticing that we are interested in the building of an optimizer, and, consequently, it is our aim to evaluate exactly the fitness function in correspondence with specified positions reached by each member of the flock. In order to address this goal, we subdivide the time axis into a sequence of time windows, each one having duration equal to a value τ. In this way, we can assume, for each ith time window TW, that the excitation (3) is a constant value equal to the one evaluated at the end of the previous $(i-1)$th TW. It follows that, for a generic TW_i that begins at the generic time t_{in} and ends at the time $t_{\mathrm{in}}+\tau$, we assume that the kth entries of the force terms (3) are equal to a constant value $\mathfrak{F}_{k,i}(t_{\mathrm{in}})$, that is, for any time value within that ith window. It is worth noticing that when $\tau \to 0$ the CFSO system would be really continuum, but the fitness function should be evaluated infinite times. Anyway, for finite values of τ, the system (5) is still time invariant within each time window. As a consequence, it is still possible to use Laplace tansforms in the previously described way. Now, the only difference is that we will obtain analytical closed forms, for the position and the velocity of each bird of the flock, valid just for a TW. The final solution on the desired time of observation will be obtained by the union of the solutions returned TW-by-TW, with the continuity being guaranteed by the initial conditions that are always evaluated at the beginning of each new TW.

2.3. Closed Forms for the Continuous-Flock-of-Starlings Optimization. On the basis of the previous assumptions, the Laplace transform related to (3), valid for the kth particle/bird, is (and clearly for the generic component jth) within the ith TW:

$$F_{k,i}(s) = \frac{\gamma g_{\mathrm{best}_i} + \lambda p_{\mathrm{best}_{k,i}}}{s}. \tag{21}$$

By elaborating and reordering (20), we obtain the following explicit expression for the Laplace transform of each kth member of the flock:

$$\begin{aligned} X_k(s) = {} & \frac{s x_k + a_k}{s^2 + (h-\omega)s + \mu} \\ & + h\frac{s^2\sum_{n=1}^{N} x_n + s\sum_{n=1}^{N} a_n + \lambda\sum_{n=1}^{N} p_{\mathrm{best}_n}}{\left(s^2 - (\omega + (N-1)h)s + \mu\right)\left(s^2 + (h-\omega)s + \mu\right)} \\ & + \frac{\lambda p_{\mathrm{best}_k}}{s\left(s^2 - (\omega-h)s + \mu\right)} \\ & + \frac{\gamma g_{\mathrm{best}}}{s\left(s^2 - (\omega + (N-1)h)s + \mu\right)}. \end{aligned} \tag{22}$$

Now it is possible to evaluate the inverse Laplace transform. Firstly, the nonzero poles of (22) are

$$s_{1,2} = \frac{\omega - h \pm \sqrt{(\omega-h)^2 - 4\mu}}{2}, \tag{23}$$

$$s_{3,4} = \frac{\omega + (N-1)h \pm \sqrt{(\omega + (N-1)h)^2 - 4\mu}}{2}. \tag{24}$$

If we assume that the parameters satisfy $(\omega-h)^2 - 4\mu \neq 0$, $(\omega + (N-1)h)^2 - 4\mu \neq 0$, all poles will have multiplicity equal to 1. Under this assumption, by applying to (22) the classic residual method for the evaluation of the inverse Laplace transforms, the time domain response is finally obtained:

$$\begin{aligned} x_k(t) = {} & x_k\sum_{m=1}^{2}\frac{(-1)^m s_m e^{s_m t}}{s_2 - s_1} + a_k\sum_{m=1}^{2}\frac{(-1)^m e^{s_m t}}{s_2 - s_1} \\ & + h\sum_{m=1}^{4}\frac{\left(\sum_{n=1}^{N} x_n\right) s_m^2 e^{s_m t}}{\prod_{n=1, n\neq m}^{4}(s_m - s_n)} \\ & + h\sum_{m=1}^{4}\frac{\left(\sum_{n=1}^{N} a_n\right) s_m e^{s_m t}}{\prod_{n=1, n\neq m}^{4}(s_m - s_n)} \\ & + h\sum_{m=1}^{4}\frac{\lambda\left(\sum_{n=1}^{N} p_{\mathrm{best}_n}\right) e^{s_m t}}{\prod_{n=1, n\neq m}^{4}(s_m - s_n)} \\ & + \lambda p_{\mathrm{best}_k}\left[\sum_{m=1}^{2}\frac{(-1)^m e^{s_m t}}{s_2 - s_1} + \frac{1}{s_1 s_2}\right] \\ & + \gamma g_{\mathrm{best}}\left[\sum_{m=1}^{2}\frac{(-1)^m e^{s_{m+2} t}}{s_{m+2}(s_4 - s_3)} + \frac{1}{s_3 s_4}\right]. \end{aligned} \tag{25}$$

As it has been already said, (25) is valid for a generic kth particle/bird just within a single time window having width τ and starting from t_{in} (i.e., $t_{\mathrm{in}} \leq t \leq t_{\mathrm{in}} + \tau$). It is the closed form which describes how the particle/bird changes its jth coordinate during time; that is, (25) describes the portion of the trajectory along the jth dimension followed by the kth member of the flock within that time window. The corresponding closed form of the velocity along the jth dimension is trivially obtainable by derivation versus time:

$$\begin{aligned} v_k(t) = {} & x_k\sum_{m=1}^{2}\frac{(-1)^m s_m^2 e^{s_m t}}{s_2 - s_1} + a_k\sum_{m=1}^{2}\frac{(-1)^m s_m e^{s_m t}}{s_2 - s_1} \\ & + h\sum_{m=1}^{4}\frac{\left(\sum_{n=1}^{N} x_n\right) s_m^3 e^{s_m t}}{\prod_{n=1, n\neq m}^{4}(s_m - s_n)} \\ & + h\sum_{m=1}^{4}\frac{\left(\sum_{n=1}^{N} a_n\right) s_m^2 e^{s_m t}}{\prod_{n=1, n\neq m}^{4}(s_m - s_n)} \end{aligned}$$

$$+ h\sum_{m=1}^{4} \frac{\lambda\left(\sum_{n=1}^{N} p_{\text{best}_n}\right) s_m e^{s_m t}}{\prod_{n=1, n\neq m}^{4} (s_m - s_n)} + \lambda p_{\text{best}_k} \sum_{m=1}^{2} \frac{(-1)^m s_m e^{s_m t}}{s_2 - s_1} + \gamma g_{\text{best}} \sum_{m=1}^{2} \frac{(-1)^m e^{s_{m+2} t}}{(s_4 - s_3)}. \tag{26}$$

These closed forms must be used as updating rules for the position and velocity of each bird at each step/time window of amplitude τ variable from a time window to another, just by using $t = \tau$. In addition the knowledge of the analytical expression related to the poles allows us to calculate the conditions on CFSO parameters, which provide the full asymptotical stability:

$$\omega < h < -\frac{\omega}{N-1} \cap \omega < 0 \cap \mu > 0. \tag{27}$$

A Matlab implementation (version 1.0) of the previously presented closed forms can be downloaded from the link in [39].

2.4. Continuous PSO versus CFSO: Effect of the Neighbour-Velocity Matching on the Collective Behaviour. The stability conditions give us the further possibility to analyze the effect of topological rules on the collective behavior of the swarm/flock members. Firstly, it is worth noticing that we can obtain the close form expressions for a continuous PSO just by posing $h = 0$ in (25) and (26), obtaining (after some substitutions)

$$x_k(t) = x_k \frac{s_2 e^{s_2 t} - s_1 e^{s_1 t}}{\sqrt{\omega^2 - 4\mu}} + a_k \frac{e^{s_2 t} - e^{s_1 t}}{\sqrt{\omega^2 - 4\mu}} + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right)\left(\frac{s_1 e^{s_2 t} - s_2 e^{s_1 t}}{\mu\sqrt{\omega^2 - 4\mu}} + \frac{1}{\mu}\right), \tag{28}$$

$$v_k(t) = x_k \frac{s_2^2 e^{s_2 t} - s_1^2 e^{s_1 t}}{\sqrt{\omega^2 - 4\mu}} + a_k \frac{s_2 e^{s_2 t} - s_1 e^{s_1 t}}{\sqrt{\omega^2 - 4\mu}} + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right)\left(\frac{e^{s_2 t} - e^{s_1 t}}{\sqrt{\omega^2 - 4\mu}}\right). \tag{29}$$

As it is possible to observe, for the continuous PSO, the way in which the velocity and the position of a generic$k$th particle (28) change during time takes into account the effects of terms which derive from the free response (x_k and v_k) and other terms which derive from the forced response (p_{best_k} and g_{best}). Thus, if we consider just the free response of the system, we will have the loss of any contribution coming from other particles (global and personal bests) since the free response of the continuous PSO depends only on the behaviour of the particle itself. On the contrary, this is not true, for the CFSO, since, in this case, the free response of a generic bird still depends on the behaviour of other members of the flock and the collective behavior exists even if we cancel the contribution of global and personal bests. These observations suggest that the collective behaviour is always present for the CFSO (and consequently for FSO), whereas the CPSO and PSO are rather based on a *forced social* behaviour due to the action of the "global" best.

2.5. CFSO Algorithm. The CFSO algorithm can be summarized by the following pseudo-code valid for a generic fitness function, $f(x^{\langle 1\rangle}, \ldots, x^{\langle D\rangle})$, to be minimized in the search space $S \subset \mathbb{R}^D$, having dimension D, with $S \equiv (x^{\langle 1\rangle}, x^{\langle 2\rangle}, \ldots, x^{\langle D\rangle})$: $x^{\langle j\rangle \min} \leq x^{\langle j\rangle} \leq x^{\langle j\rangle \max}$ for $j = 1, \ldots, D$.

Initialization

Set:

(i) n_{birds} is the total number of elements into the flock;

(ii) number of birds into the flock controlled by one single bird, $N_{\text{ctrl\_birds}}$.

(iii) Inertial coefficient ω, cognitive coefficient, λ, social coefficient, γ, and topological coefficient h (i.e., choice of the typology of the poles)

(iv) Maximum number of TWs, $N_{\text{step\_max}}$;

(v) Velocities for each kth bird $v_k^j(t = 0) = \text{random}(-1, 1) \cdot V_{\max}$;

(vi) Position $(x_k^{\langle 1\rangle}(0), \ldots, x_k^{\langle D\rangle}(0))$ of each kth bird

(vii) Initial personal fitness $f_{p_j}(t = 0)$;

(viii) Initial global fitness $g(t = 0)$;

(ix) Fitness threshold goal_fitness = arbitrary small;

Main Loop

For each kth particle and for TW of amplitude τ, until we reach the assigned maximum number of TW,

(i) update, for each kth particle, position and velocity at end of considered TW by using (25) and (26) computed for $t = \tau$. (let us indicate with $(x_k^{\langle 1\rangle}, \ldots, x_k^{\langle D\rangle})$ and $(v_k^{\langle 1\rangle}, \ldots, v_k^{\langle D\rangle})$ respectively the position and the velocity at the end of considered TW)

(ii) evaluate fitness $f_k = f(x_k^{\langle 1\rangle}, \ldots, x_k^{\langle D\rangle})$

(iii) If f_k is better than the personal best fitness of the kth particle f_{p_k}, then assign current position as personal best position and update the personal best fitness:

$$p_{\text{best}_k}^{\langle j\rangle} = x_k^{\langle j\rangle} \quad \forall j\text{th dimension}, \qquad f_{p_k} = f_k \tag{30}$$

(iv) If f_k is better than global best fitness, then assign current position as global best position and update the global best fitness:

$$g_{\text{best}}^{\langle j\rangle} = x_k^{\langle j\rangle} \quad \forall j\text{th dimension}$$
$$g(t) = f_k \tag{31}$$

(v) Assign in new initial conditions $(x_k^{\langle 1\rangle}, \ldots, x_k^{\langle D\rangle})$, $(v_k^{\langle 1\rangle}, \ldots, v_k^{\langle D\rangle})$ and excitations $(p_{\text{best}_k}^{\langle 1\rangle}, \ldots, p_{\text{best}_k}^{\langle D\rangle})$ $(k = 1, \ldots, n_{\text{birds}})$ and $(g_{\text{best}}^{\langle 1\rangle}, \ldots, g_{\text{best}}^{\langle D\rangle})$

End for

It is worth noticing again that for each TW the excitation is constant values and from a TW to the successive the continuity of the trajectories is guaranteed by means of the initial conditions, in terms of positions and velocities, evaluated as the last values related to the previous TW.

2.6. Assignments of Poles for Different Flock Behaviours. The individuals of a swarm can be driven through three different type of trajectories for three different assignments of the poles: (1) couple of conjugate pure imaginary poles (stable system); (2) poles with real part negative (asymptotically stable system); and (3) poles with real part positive (unstable system). With the aim to investigate deeper on the free response of the CFSO, let us assume the global and all personal bests to be equal to constant values for the whole duration of the simulation. This last assumption allows to investigate the free response since no variation for global as well for personal bests is possible. For the sake of simplicity we refer to a bidimensional space since in this way we have the possibility to plot the trajectories followed by the birds.

2.6.1. Couple of Conjugate Pure Imaginary Poles. In this case, by the previous assumption, the forced terms are not allowed to change from a TW to another one; we can look at the trajectory after the end of the transient phase. Since we have imposed a couple of conjugate pure imaginary poles for at least one member of the swarm, the relative components of both position and velocity show a periodic behaviour. Consequently each bird having these kinds of poles will describe (in 2D case) a closed loop as Figure 1 shows. In terms of optimum searching this kind of trajectory can be interpreted as a path made by one member which explores the surrounding space, in order to search for a possible new better solution.

2.6.2. Poles with Negative Real Part (Asymptotically Stable System). Clearly the previously described trajectories are not useful from the viewpoint of accuracy in detecting optimum solutions. Thus, in order to refine the values of candidate solutions which could lie in a subspace investigated by members with conjugate pure-imaginary poles, the values of these poles are converted by setting their real part as negative. Indeed, according to (25) and (26), when the transient ends the positions are distributed around the global best and all the velocities have null values. Moreover, during the transient phase, the birds follow convergent trajectories towards the global best (see Figure 2), and consequently a refinement of the solution is obtainable.

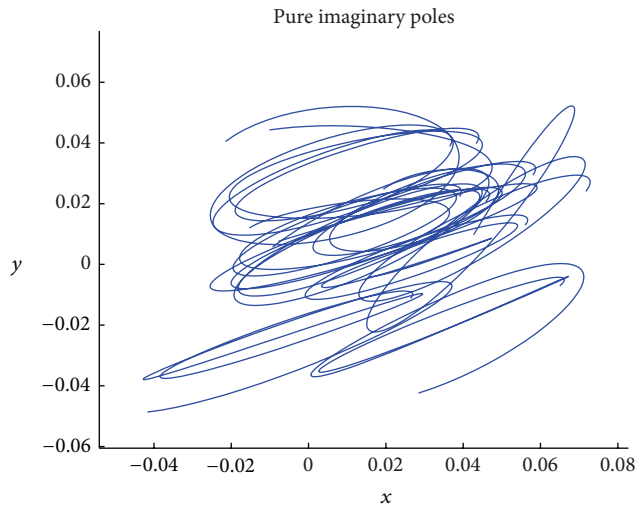

Figure 1: Typical trajectories of birds in the flock for pure imaginary poles. It is possible to see the 30 birds starting around the origin, making 30 closed loops.

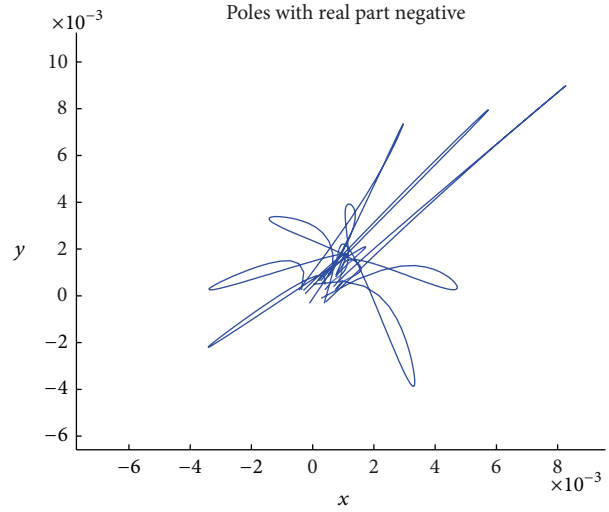

Figure 2: Typical trajectories of birds in the flock for asymptotically stable poles. The convergence in the region containing global best is evident that is coincident with the origin of the axes. The effect of refinement can be seen by comparing the values of x and y with those of the previous Figure 1.

2.6.3. Poles with Positive Real Part (Unstable System). The last case presented herein is the one concerning poles with positive real part. As previously stated for pure imaginary poles, it is possible that in some situation the birds performing CFSO have fallen in a region containing a local minimum. It is important to have a strategy for escaping from that situation for restarting the exploration. This can be easily done by

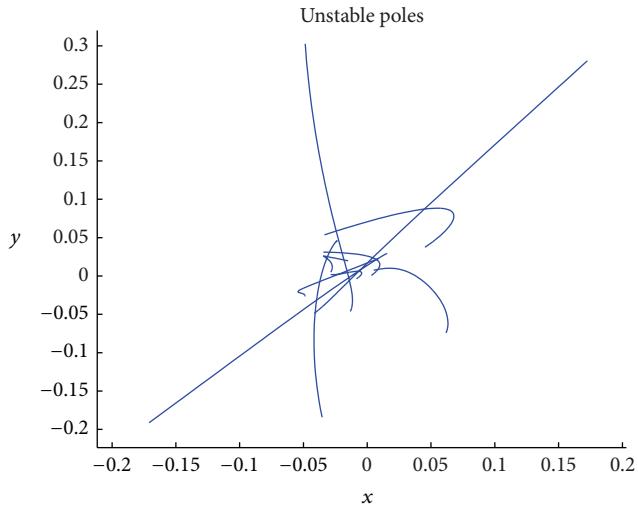

Figure 3: Typical trajectories of birds in the flock for unstable poles. The trajectories of 10 birds, starting also in this case around the origin, diverge after few steps, and consequently they escape from minimum.

choosing poles with a positive real part. Indeed, in this case the trajectories will be forced to diverge also in presence of a local minimum (see Figure 3). In addition if the poles have also an imaginary part other than zero, the oscillatory behaviour is combined with the exponential and then they will escape from the entrapping region by following a sort of spiral (unstable focus). Clearly this kind of poles could lead the birds to exit from domain; thus they must be used only for few TWs (eventually just one TW if we choose at the same time a high value for TW amplitude, τ, as will be discussed in the next section).

2.7. Role of the τ Parameter. As previously stated, an important feature of the CFSO is the possibility to predict the behaviour of particles/birds simply studying the poles of Laplace transforms for different TWs. Indeed, by changing these poles we can force the trajectories of the swarm members to converge towards a point, to escape from a possible local minimum, or to stay into a limited zone (closed loops) of the solution space. In this way we can improve the capabilities of exploration and exploitation simply by changing the poles of state equations system at each time window. Thus, by a suitable choice of the poles from a TW to another one, it is possible to control the whole behaviour of the swarm members. On the contrary, the noncontinuous but simply numerical algorithms employ the random updating of parameters to avoid local minima. By using the closed forms of CFSO, the randomness of parameters can be eliminated because the user *a priori* knows how he must manage the parameters to obtain convergence or divergence, that is, how to exalt the exploitation rather than the exploration or *vice versa*. In addition, at any time during the optimization procedure, the CFSO can refine the found solution through a suitable decreasing of the TWs amplitudes, as it will be discussed in the Validation. Indeed, as shown in Figure 4, it is always possible to refine the solution simply by using small TW amplitudes together with negative real part poles: small TWs return small displacements (consequently a large number of personal bests is investigated), and the refining of values of the candidate solutions is obtainable. In the same way, large TW amplitudes together with positive real part poles are useful to quickly escape from local minima.

3. From CFSO to CFSO^3

On the basis of the previous remarks on CFSO, for enhancing both exploration and convergence, a hybridization strategy similar to the one successfully followed in [40] has been developed. In [40] three different numerical heuristics have been used in such way to exalt their different abilities in exploration, or exploitation or refinement. In the present case, the same idea has been applied to take advantage from the different behaviours provided by CFSO depending on the different setting of its parameters (i.e., the nature of the poles). In particular, at the beginning of the whole optimization process, the CFSO is implemented by setting *pure imaginary poles* (CFSO*pi*). The CFSO*pi* performs exploration and whenever it finds a subregion in which there is a high probability of discovering a global minimum (briefly called "*suspected region*" according to [40]), two operations are made in cascade: (1) the exploitation and the refinement of the solution, by launching a *asymptotically stable* CFSO (CFSO*as*), that is, a CFSO, in which the real part of poles is set to be positive; in such way we can exalt the exploitation of the "*suspected region*" for finding the value of the minimum that lies here (the CFSO population is initialized by means of the result that the CFSO*pi* had just found); (2) the CFSO poles are then temporarily changed in *unstable poles* (CFSO*us*) for N steps (after which pure *imaginary poles* CFSO is reset), for escaping from the minimum inside the exploited *suspected region* (since it could be a local minimum). This hybridization has been called CFSO^3 in order to underline the three different aspects of CFSO which have been used in synergy. In other words, the optimization process is fully supervised; that is, the change of the nature of poles is *a priori* planned by a criterion based on monitoring the value of the cost function. In the setup of the procedure no stochastic parameter has been used with the exception of the starting positions and velocities of the birds. This means that if one launches CFSO^3 starting from the same initial conditions, it will obtain the same identical results.

3.1. Parallel Architecture. CFSO^3 provides its best performances on a distributed architecture, and the algorithm can be fully designed for a master-slave configuration for parallel computation [41, 42]. According to the different CFSO peculiarities related to different setting of the poles, CFSO^3 uses imaginary poles and unstable poles just on the master node, whereas the refinement phase with asymptotically stable poles is performed on slave ones. Virtually, the CFSO*pi* performs exploration of the solution space on the Mater node and whenever it finds a "*suspected region,*" two simultaneous operations are made:

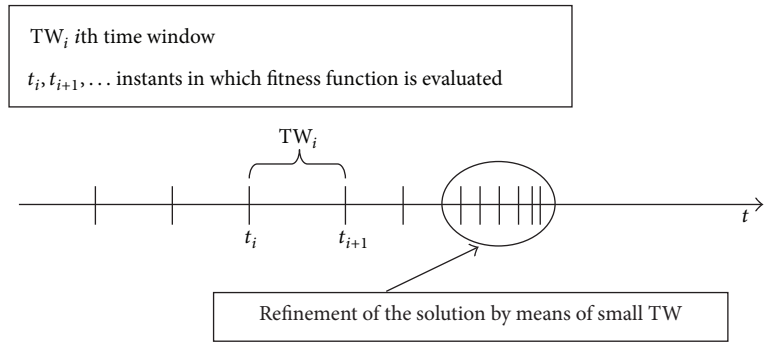

Figure 4: Example of managing of time window amplitudes for refining candidate solution.

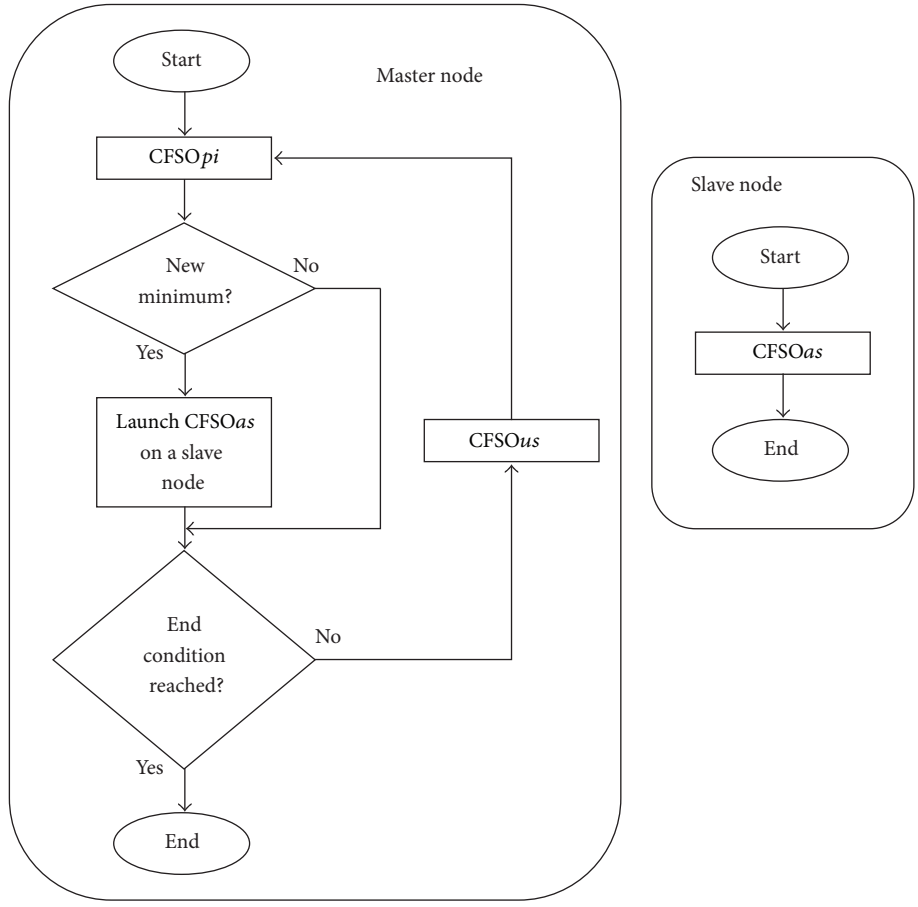

Figure 5: Flow chart of parallel version of CFSO^3.

(1) CFSO*us* is launched for N steps (i.e., N TWs) on master node (thereafter *imaginary poles* are reset and we return to the CFSO*pi*), in such a way to escape from the "*suspected region*" and to restart exploration;

(2) CFSO*as* is launched on the first available slave node of the cluster with the aim to refine the solution into that "*suspected region*," the population of the CFSO*as* with being initialized on the basis of the best result found by CFSO*pi* at the current iteration.

In more detail, the CFSO*as* will be left to run for a fixed number of iterations $N_{\text{refinement-max}}$ with a duration of TW (parameter τ) around 10 times smaller than the one used in CFSO*pi* on the master node. While the processes on the slave nodes are running, the CFSO*pi* is provisionally substituted by the CFSO*us* for N (generally not more than a couple) TWs; thereafter the CFSO*pi* is relaunched. CFSO*pi* still continues to explore the solution space on the Master node and whenever it finds a further "*suspected region*," another CFSO*as* is launched on a different slave node and so on. After having delivered the found solution to the master, the slave node is ready to be utilized for a further exploration on a new "*suspected region*." All final results coming from slave nodes are stored in a dedicated file that is located in a shared folder within the master node. Finally, when all processes

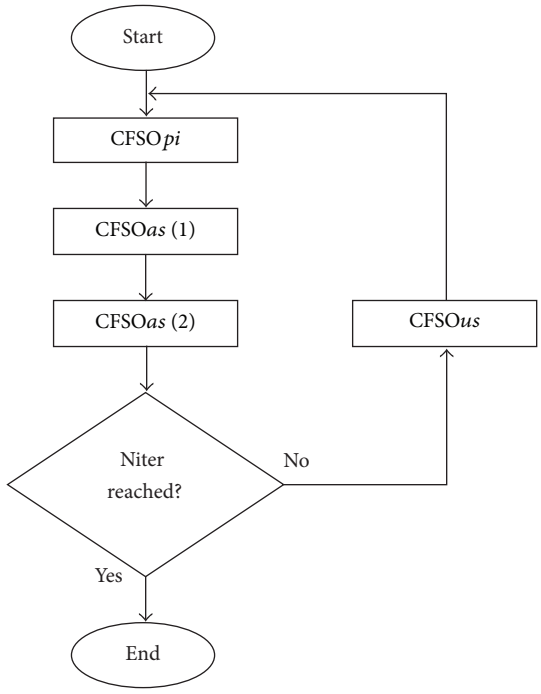

Figure 6: Flow chart of the CFSO^3 implementation used for tests.

Table 1: Parameters used in CFSO^3.

	ω	λ	γ	h	τ
CFSO*pi*	−0.8147	0.421	0.579	9.502*e*−02	0.2
CFSO*as* (1)	−2.126	0.482	0.913	−0.668	0.05
CFSO*as* (2)	−2.126	0.482	0.913	−0.668	0.02
CFSO*us*	0.126	0.482	0.413	−0.668	0.02

Table 2: Success rate (%) of CFSO and CFSO^3 for different number of cycles.

	CFSO (10000 FEs)	CFSO^3 $N_{\text{iter}} = 1$; (3000 FEs)	CFSO^3 $N_{\text{iter}} = 2$; (7000 FEs)	CFSO^3 $N_{\text{iter}} = 3$; (11000 FEs)
Shifted sphere function	30	48	86	97
Schwefel's problem 2.21	36	42	80	98
Shifted Rosenbrock's function	30	46	84	97
Shifted Rastrigin's function	15	24	63	86
Shifted Griewank's function	21	42	81	92
Shifted Ackley's function	23	22	56	70

have been concluded (i.e., all PC slaves have completed the refinement procedures previously described, and the CFSO*pi* on the master node has ended all the programmed iterations), a list of all results coming from slave nodes will be available within the master node storage file. Then, the best minimum (global optimum) is trivially identified from that list. The flow chart of the parallel architecture is shown in Figure 5.

4. Validation

The CFSO algorithm has been successfully validated for the solution of engineering design problems [33, 34]. Moreover, it has also been compared with the performances coming from a generic implementation of PSO for the solution of classical optimization benchmarks. It is worth noticing that all validations have been made by starting from the same initial conditions (positions and velocities) [35] with the aim of obtaining the same starting situations as for CFSO, and standard PSO.

Hereafter we follow a validation approach of the CFSO^3 by considering the same tests presented in [35] and other tests on typical benchmarks [43, 44]. In particular, we have used the 2D version of the seven benchmarks proposed in [44]: F1 Shifted Sphere Function, F2 Schwefel's Problem 2.21, F3 Shifted Rosenbrock's Function, F4 Shifted Rastrigin's Function, F5 Shifted Griewank's Function, F6 Shifted Ackley's Function, and the FastFractal "DoubleDip" function, from the fractal function benchmarking suite [44]. The choice of shifted version of benchmarks and the DoubleDip function has the aim to mitigate the effect of possible initialization biases. In particular, we compare the results obtained by CFSO in full connected standard implementation (that was already successfully compared with PSO in [35]) with those obtainable by the present CFSO^3, implemented by following the flow chart described in Figure 6, that is, without the use of the parallel architecture. In practice, we repeat N_{iter} times a cycle in which a CFSO*pi*, two CFSO*as*, and CFSO*us* are sequentially executed, each one for 100 steps and by using the parameters reported in Table 1 (in all cases $n_{\text{birds}} = 10$). It is important to note that the deterministic nature of these algorithms causes the identical repetition of trajectories if the same set of guess values is used.

In order to adopt the same initial conditions for all cases, we have used the same positions and velocities as in [35] (100 tests for each benchmark). In particular, we have used 1000 steps for CFSO (the number of steps was the stop criterion adopted), which correspond to an amount of function evaluations (FEs) equal to 10000 for CFSO, whereas for CFSO^3 the number of steps and FEs depend on N_{iter} according to the expression $N_{\text{steps}} = 300 + 400 \times (N_{\text{iter}} - 1)$ and FEs $= 10 \times N_{\text{steps}}$. The performances for benchmarks (F1–F6) are presented in Table 2 in terms of success rate (% of success), that is, the percentage of tests, for which the error achieved is less than a prefixed threshold (chosen at 10^{-6} for F1, F3, and F5 and 10^{-4} for F2, F4, and F6).

From Table 2 it is possible to see that the success rate increases by increasing the number of cycles executed (success rate around 70% and 97% even after 3 cycles for these examples). Clearly if we use $N_{\text{iter}} = 1$, practically we make the refinement but we do not use CFSO*us* to escape from possible local minima. However, in this case it is also possible to appreciate an improvement of the performance. These results confirm that, thanks to the refinement approach obtained

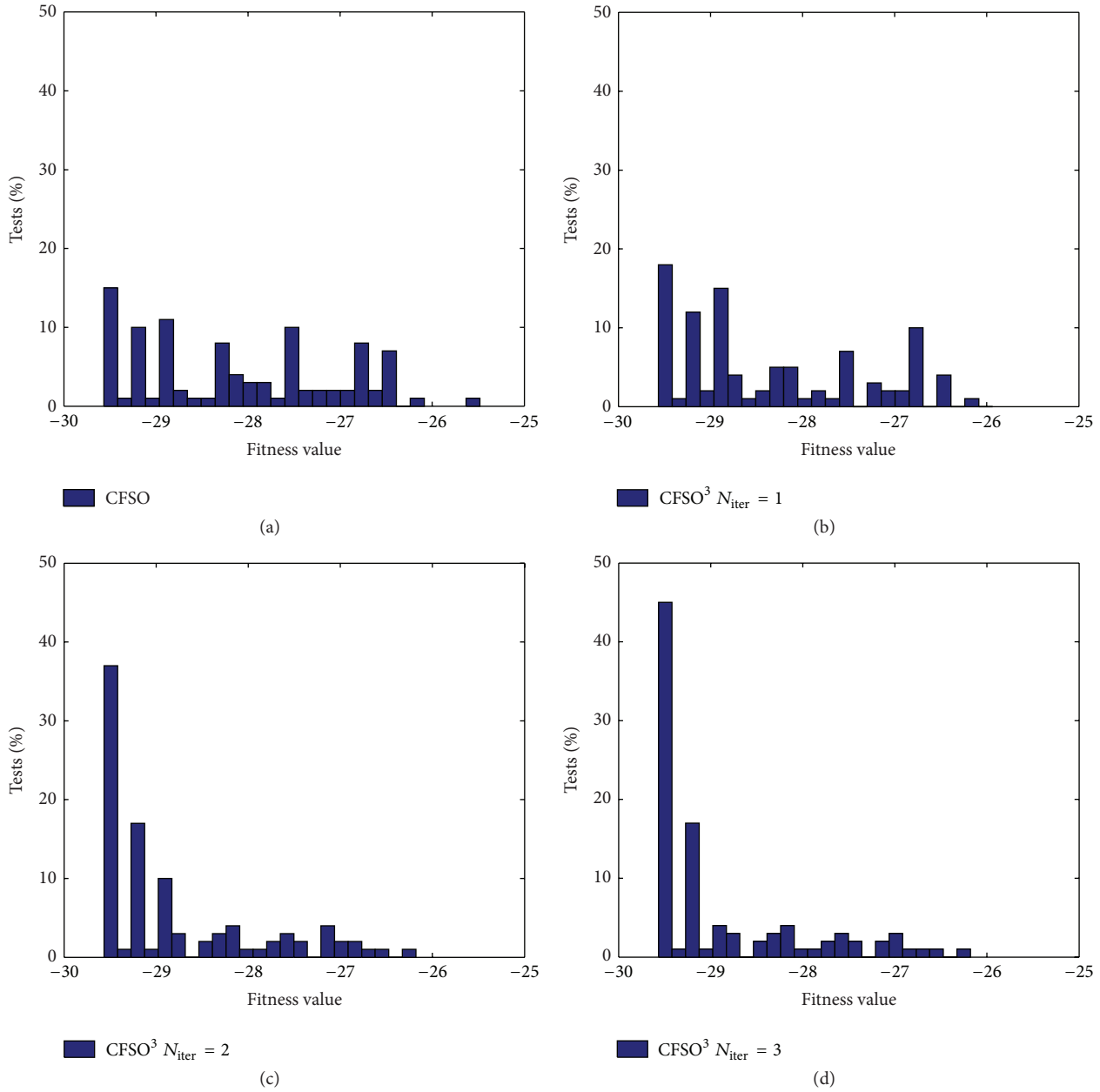

Figure 7: The % of tests giving correspondent fitness values for the DoubleDip function.

by CFSO*as* and to the capability of escaping from local minima given by CFSO*us*, we are able to find the solution of an optimization problem almost independently by the initial conditions imposed, that is, without using any stochastic parameters, but rather by a supervised optimization approach. In addition, it is worth noticing that the maximum error achieved after 3 cycles is of few percents, for all the benchmarks.

For the DoubleDip function, since we do not have any information about the values of the global minimum, we present the results by using a set of histograms (shown in Figure 7) representative of the % of tests giving the relative fitness values. As can be seen in Figure 7, the larger the number of iterations, the larger the number of tests giving a minimum percentage value around to 29.56 (the minimum value obtained among all tests is 29.56615 with respect to 29.5641 obtained in [35]).

Clearly, as previously stated, these results are just early validations, whereas further deeper investigations must be done in order to evaluate the effectiveness of such modification and make the setup of the continuous optimizer more efficient. For example, the nature of the benchmarks F2, F4, and F6, which present a higher gradient, should suggest to increase the number of steps devoted to refinement or to use smaller values for τ (in order not to make any change in the code used for the benchmark, the threshold was kept in this case at 10^{-4}). Nevertheless the promising results obtained in validation confirm the goodness of the proposed supervised deterministic approach. In addition, with the aim of involving the scientific community in the use

and validation of the developed CFSO code, we have provided its MATLAB version 1.0 in [39].

5. Conclusions

In this paper a supervised approach based on the hybridization of different CFSO, a heuristics that converts numerical swarm-based algorithms into analytical closed forms, has been presented. The use of the closed forms allows investigating the different types of trajectories that can be performed by swarm members by different parameter setting; that is, what is the influence of the values of parameters on the divergence or the convergence (exploration or exploitation) of the swarm-based algorithms? On the other hand, the development of the CFSO^3 has allowed eliminating the random updating of parameters, a common praxis present in swarm-based algorithms: CFSO^3 manages the parameters switching from a behaviour to another. All these items allow us to design a supervised optimizer, which combines exploration, escaping from minima, and refinement abilities (resp.: CFSO*pi*, CFSO*us*, and CFSO*as*). CFSO^3 can be supported by both sequential and parallel implementations. Although further investigations would be necessary for a deeper analysis of this optimizer, the promising results returned from validations confirm the effectiveness of the proposed approach (a CFSO Matlab code is available in [39]). In addition the CFSO/CFSO^3 equations, defining a continuous dynamic system, can be also hardware implemented by means of an analogical circuit, giving the possibility to put in practice new typology optimizers [34], suitable for the solution of real-time problems also in combination with hybrid artificial intelligence systems which use neural networks [45, 46].

Appendix

PSO as Approximation of CPSO Equations

It is interesting to note that the numerical rules of PSO can be obtained by (28) and (29) under some appropriate approximation. Let us start with (28) and rewrite it for $t = \tau = 1$:

$$x_k(1) = x_k \frac{s_2 e^{s_2} - s_1 e^{s_1}}{\sqrt{\omega^2 - 4\mu}} + a_k \frac{e^{s_2} - e^{s_1}}{\sqrt{\omega^2 - 4\mu}} + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right)\left(\frac{s_1 e^{s_2} - s_2 e^{s_1}}{\mu\sqrt{\omega^2 - 4\mu}} + \frac{1}{\mu}\right). \tag{A.1}$$

By substituting the first-order approximation for the exponential

$$e^s \approx 1 + s,$$

$$\begin{aligned} x_k(1) &\approx x_k \frac{s_2(1+s_2) - s_1(1+s_1)}{\sqrt{\omega^2 - 4\mu}} + a_k \frac{(1+s_2) - (1+s_1)}{\sqrt{\omega^2 - 4\mu}} \\ &\quad + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right) \\ &\quad \times \left(\frac{s_1(1+s_2) - s_2(1+s_1)}{\mu\sqrt{\omega^2 - 4\mu}} + \frac{1}{\mu}\right) \\ &= x_k \frac{(s_2 - s_1)(1 + (s_2 + s_1))}{\sqrt{\omega^2 - 4\mu}} + a_k \frac{s_2 - s_1}{\sqrt{\omega^2 - 4\mu}} \\ &\quad + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right)\left(\frac{s_1 - s_2}{\mu\sqrt{\omega^2 - 4\mu}} + \frac{1}{\mu}\right) \\ &= x_k(1 + (s_2 + s_1)) + a_k = x_k(1 + \omega) + a_k \\ &= x_k + x_k\omega - \omega x_k + u_k = x_k + v_k, \end{aligned} \tag{A.2}$$

that is, the PSO updating rule for positions.

In an analogous way it is possible to retrieve the PSO rule for updating velocity (29):

$$\begin{aligned} v_k(1) &= x_k \frac{s_2^2 e^{s_2} - s_1^2 e^{s_1}}{\sqrt{\omega^2 - 4\mu}} + a_k \frac{s_2 e^{s_2} - s_1 e^{s_1}}{\sqrt{\omega^2 - 4\mu}} \\ &\quad + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right)\left(\frac{e^{s_2} - e^{s_1}}{\sqrt{\omega^2 - 4\mu}}\right) \\ &\approx x_k \frac{s_2^2(1+s_2) - s_1^2(1+s_1)}{\sqrt{\omega^2 - 4\mu}} \\ &\quad + a_k \frac{s_2(1+s_2) - s_1(1+s_1)}{\sqrt{\omega^2 - 4\mu}} \\ &\quad + \left(\lambda p_{\text{best}_k} + \gamma g_{\text{best}}\right)\left(\frac{1 + s_2 - 1 - s_1}{\sqrt{\omega^2 - 4\mu}}\right) \\ &= \cdots \\ &= v_k(1 + \omega) + \left(\lambda\left(p_{\text{best}_k} - x_k\right) + \gamma\left(g_{\text{best}} - x_k\right)\right). \end{aligned} \tag{A.3}$$

From this last equation, it is clear that, in order to obtain the correct numerical version of PSO, an equivalence relation must be established between the inertial parameter ω_{pso} of the PSO algorithm and the inertial parameter ω_{cpso} of the CPSO:

$$\omega_{\text{pso}} = \omega_{\text{cpso}} + 1. \tag{A.4}$$

References

[1] F. Riganti Fulginei and A. Salvini, "Comparative analysis between modern heuristics and hybrid algorithms," *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 26, no. 2, pp. 259–268, 2007.

[2] C. W. Liew and M. Labiri, "Exploration or convergence? Another metacontrol mechanism for Gas," in *Proceedings of*

the 18th International Florida AI Research Society Conference, pp. 251–256, May 2005.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Perth, Australia, December 1995.

[4] C. W. Reynolds, "Flocks, herds and schools: a distributed behavioral model," *Computer Graphics*, vol. 21, no. 4, pp. 25–34, 1987.

[5] F. Heppner and U. Grenander, "A stochastic nonlinear model for coordinated bird flocks," in *The Ubiquity of Chaos*, S. Krasner, Ed., AAAS Publications, Washington, DC, USA, 1990.

[6] A. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, Wiley, New York, NY, USA, 2005.

[7] M. Clerc, *Particle Swarm Optimization*, ISTE, London, UK, 2006.

[8] M. R. AlRashidi and M. E. El-Hawary, "A survey of particle swarm optimization applications in electric power systems," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 913–918, 2009.

[9] S. Coco, A. Laudani, G. Pollicino, G. Pulcini, F. Riganti Fulginei, and A. Salvini, "TWT magnetic focusing structure optimization by parallel evolutionary algorithm," *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 31, no. 5, pp. 1338–1346, 2012.

[10] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, May 1998.

[11] J. H. Seo, C. H. Im, S. Y. Kwak, C. G. Lee, and H. K. Jung, "An improved particle swarm optimization algorithm mimicking territorial dispute between groups for multimodal function optimization problems," *IEEE Transactions on Magnetics*, vol. 44, no. 6, pp. 1046–1049, 2008.

[12] K. Parsopoulos and M. Vrahatis, "Modification of the particle swarm optimizer for locating all the global minima," in *Artificial Neural Networks and Genetic Algorithms*, pp. 324–327, Springer, 2001.

[13] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proceedings of IEEE Swarm Intelligence Symposium (SIS '07)*, pp. 120–127, IEEE Press, April 2007.

[14] X. Chen and Y. Li, "A modified PSO structure resulting in high exploration ability with convergence guaranteed," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 37, no. 5, pp. 1271–1289, 2007.

[15] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 39, no. 6, pp. 1362–1381, 2009.

[16] M. M. Ali and P. Kaelo, "Improved particle swarm algorithms for global optimization," *Applied Mathematics and Computation*, vol. 196, no. 2, pp. 578–593, 2008.

[17] M. A. Montes de Oca, T. Stützle, M. Birattari, and M. Dorigo, "Frankenstein's PSO: a composite particle swarm optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1120–1132, 2009.

[18] Z. Xinchao, "A perturbed particle swarm algorithm for numerical optimization," *Applied Soft Computing*, vol. 10, no. 1, pp. 119–124, 2010.

[19] L. Liu, S. Yang, and D. Wang, "Particle swarm optimization with composite particles in dynamic environments," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 40, no. 6, pp. 1634–1648, 2010.

[20] R. Mendes, *Population Toplogies and Their Influence in Particle Swarm Performance [Ph.D. thesis]*, Universidade do Minho, Braga, Portugal, 2004.

[21] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[22] J. Kennedy and R. Mendes, "Neighborhood topologies in fully informed and best-of-neighborhood particle swarms," *IEEE Transactions on Systems, Man, and Cybernetics C*, vol. 36, no. 4, pp. 515–519, 2006.

[23] A. S. Mohais, R. Mendes, C. Ward, and C. Posthoff, "Neighborhood re-structuring in particle swarm optimization," in *Proceedings of the 18th Australian Joint Conference on Artificial Intelligence*, vol. 3809 of *Lecture Notes in Computer Science*, p. 776, Sydney, Australia, December 2005.

[24] H. Liu, E. Howley, and J. Duggan, "Particle swarm optimisation with gradually increasing directed neighbourhoods," in *Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference (GECCO '11)*, pp. 29–36, July 2011.

[25] M. A. Montes de Oca, T. Stützle, K. van den Enden, and M. Dorigo, "Incremental social learning in particle swarms," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 41, no. 2, pp. 368–384, 2011.

[26] X. Li, "Niching without niching parameters: particle swarm optimization using a ring topology," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, 2010.

[27] F. Riganti Fulginei and A. Salvini, "Hysteresis model identification by the flock-of-starlings optimization," *International Journal of Applied Electromagnetics and Mechanics*, vol. 30, no. 3-4, pp. 321–331, 2009.

[28] F. Riganti Fulginei and A. Salvini, "The flock-of starlings optimization: influence of topological rules on the collective behaviour of swarm intelligence," in *Computational Methods for the Innovative Design of Electrical Devices*, Studies in Computational Intelligence, pp. 139–157, Springer, 2010.

[29] M. Ballerini, N. Cabibbo, R. Candelier et al., "Interaction ruling animal collective behavior depends on topological rather than metric distance: evidence from a field study," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 4, pp. 1232–1237, 2008.

[30] J. L. Fernández-Martínez, E. García-Gonzalo, and J. P. Fernández-Alvarez, "Theoretical analysis of particle swarm trajectories through a mechanical analogy," *International Journal of Computational Intelligence Research*, vol. 4, no. 2, pp. 93–104, 2008.

[31] H. M. Emara and H. A. Abdelfatah, "Continuous swarm optimization technique with stability analysis," in *Proceedings of the American Control Conference (AAC '04)*, vol. 3, pp. 2811–2817, July 2004.

[32] S. M. Mikki and A. A. Kishk, "Physical theory for particle swarm optimization," *Progress in Electromagnetics Research*, vol. 75, pp. 171–207, 2007.

[33] S. Coco, A. Laudani, F. Riganti Fulginei, and A. Salvini, "Accurate design of Helmholtz coils for ELF Bioelectromagnetic interaction by means of continuous FSO," *International Journal of Applied Electromagnetics and Mechanics*, vol. 39, no. 1–4, pp. 651–656, 2012.

[34] A. Laudani, G. Pulcini, F. Riganti Fulginei, and A. Salvini, "Swarm circuits performing optimization and inverse problems," in *Proceedings of the International Workshops on Optimization and Inverse Problems in Electromagnetism (OIPE '12)*, Ghent, Belgium, September 2012.

[35] A. Laudani, F. Riganti Fulginei, and A. Salvini, "Closed forms for the fully-connected continuous flock-of-starlings optimization algorithm," in *Proceedings of the 15th International Conference*

on Computer Modeling and Simulation (UKSim '13), Cambridge, UK, April 2013.

[36] W. Li, "Stability analysis of swarms with general topology," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 38, no. 4, pp. 1084–1097, 2008.

[37] V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming, "Stability analysis of the particle dynamics in particle swarm optimizer," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 245–255, 2006.

[38] S. Kiranyaz, T. Ince, A. Yildirim, and M. Gabbouj, "Fractional particle swarm optimization in multidimensional search space," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 40, no. 2, pp. 298–319, 2010.

[39] CFSO—Continuous Flock-of-Starlings Optimization Algorithm tool, 2013, http://www.dea.uniroma3.it/elettrotecnica/.

[40] F. Riganti Fulginei, A. Salvini, and G. Pulcini, "Metric-topological-evolutionary optimization," *Inverse Problems in Science and Engineering*, vol. 20, no. 1, pp. 41–58, 2012.

[41] S. Coco, A. Laudani, G. Pulcini, F. Riganti Fulginei, and A. Salvini, "Shape optimization of multistage depressed collectors by parallel evolutionary algorithm," *IEEE Transactions on Magnetics*, vol. 48, no. 2, pp. 435–438, 2012.

[42] S. Coco, A. Laudani, F. Riganti Fulginei, and A. Salvini, "TEAM problem 22 approached by a hybrid artificial life method," *The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 31, no. 3, pp. 816–826, 2012.

[43] C. MacNish, "Towards unbiased benchmarking of evolutionary and hybrid algorithms for real-valued optimisation," *Connection Science*, vol. 19, no. 4, pp. 361–385, 2007.

[44] K. Tang, X. Yao, P. N. Suganthan et al., "Benchmark functions for the CEC'2008 special session and competition on large scale global optimization," Tech. Rep., Nature Inspired Computation and Applications Laboratory, USTC, China, 2007.

[45] M. Gneo, M. Schmid, S. Conforto, and T. D'Alessio, "A free geometry model-independent neural eye-gaze tracking system," *Journal of NeuroEngineering and Rehabilitation*, vol. 9, article 82, 2012.

[46] I. Bernabucci, S. Conforto, M. Capozza, N. Accornero, M. Schmid, and T. D'Alessio, "A biologically inspired neural network controller for ballistic arm movements," *Journal of NeuroEngineering and Rehabilitation*, vol. 4, article 33, 2007.