International Journal of Computer Networks and Communications Security

VOL. 8, NO. 2, February 2020, 17–25
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

A Novel Hybrid Dragonfly Algorithm with Modified Conjugate Gradient Method

Layth Riyadh Khaleel[1] and Prof. Dr. Ban Ahmed Mitras[2]

[1] M.sc. Student, department of mathematics, college of computer sciences & mathematics, Mosul University

[2] Prof. Dr. department of mathematics, college of computer sciences & mathematics, Mosul University

[1]laythmath10@gmail.com, [2]dr.banah.mitras@gmail.com

ABSTRACT

Dragonfly Algorithm (DA) is a meta-heuristic algorithm, It is a new algorithm proposed by Mirjalili in (2015) and it simulate the behavior of dragonflies in their search for food and migration. In this paper, a modified conjugate gradient algorithm is proposed by deriving new conjugate coefficient. The sufficient descent and the global convergence properties for the proposed algorithm are proved. Novel hybrid algorithm of the dragonfly (DA) was proposed with modified conjugate gradient Algorithm which develops the elementary society that is randomly generated as the primary society for the dragonfly optimization algorithm using the characteristics of the modified conjugate gradient algorithm. The efficiency of the hybrid algorithm was measured by applying it to (10) of the optimization functions of high measurement with different dimensions and the results of the hybrid algorithm were very good in comparison with the original algorithm.

Keywords: *Conjugate Gradient Methods, Meta-Heuristic Algorithms, Dragonfly Optimization Algorithm*.

1 INTRODUCTION

Optimization can be defined as one of the branches of knowledge dealing with discovering or arriving at the optimal solutions to a specific issue within a set of alternatives.[1] The methods of solving optimization problems are divided into two types of algorithms: Deterministic Algorithms and Stochastic Algorithms [2]. Most of classical algorithms are specific algorithms. For example, the Simplex method in linear programming is a specific algorithm, and some specific algorithms use tilt information (Gradient), which is called slope-based algorithms. For example, Newton-Raphson algorithm) is an algorithm based on slope or derivative [3]. As for random algorithms, they have two types of algorithms, although the difference between them is small: Heuristic Algorithms and Meta-Heuristic Algorithms. In 2015 a new algorithm was proposed by Mirjalili which is a dragonfly algorithm that simulates the behavior of dragonflies in their search for food and migration [4]. In 2016, Bashishtha and Srivastava used a dragonfly algorithm to address the problem of optimal energy flow in an electric power system. [5]. In the same year, Pathania and others used a dragonfly algorithm to solve the issue of multi-target distribution of the thermal system [6]. In 2017, Abhiraj and Aravindhababu used a dragonfly algorithm to reconfigure distribution networks in order to improve the electrical potential winding [7].

The aim of the research is: First: modified conjugate gradient method is proposed by deriving a new conjugancy coefficient named MCG algorithm.

Second: proposed a new hybrid algorithm consisting of a dragonfly algorithm (DA) with modified Conjugate Gradient conjugation methods called the DA-MCG algorithm.

2 CONJUGATE GRADIENT METHOD

In unconstrained optimization, we minimize an objective function which depends on real variables with no restrictions on the values of these variables. The unconstrained optimization problem is:

$$Min \quad f(x): x \in R^n \qquad (1),$$

where $f : R^n \to R$ is a continuously differentiable function, bounded from below. A nonlinear conjugate gradient method generates a sequence $\{x_k\}$, k is integer number, $k \geq 0$. Starting from an initial point x_0, the value of x_k calculate by the following equation:

$$x_{k+1} = x_k + \lambda_k d_k; \qquad (2),$$

where the positive step size $\lambda_k > 0$ is obtained by a line search, and the directions d_k are generated as:

$$d_{k+1} = -g_{k+1} + \beta_k d_k; \qquad (3),$$

Where, $d_0 = -g_0$, the value of β_k is determine according to the algorithm of Conjugate Gradient (CG), and its known as a conjugate gradient parameter, $s_k = x_{k+1} - x_k$ and $g_k = \nabla f(x_k) = f'(x_k)$, consider ‖.‖ is the Euclidean norm and $y_k = g_{k+1} - g_k$. The termination conditions for the conjugate gradient line search are often based on some version of the Wolfe conditions. The standard Wolfe conditions

$$f(x_k + \lambda_k d_k) - f(x_k) \leq \rho \lambda_k g_k^T d_k \qquad (4)$$

$$g(x_k + \lambda_k d_k)^T d_k \geq \sigma g_k^T d_k; \qquad (5);$$

where d_k is a descent search direction and $0 < \rho < \sigma < 1$, where β_k is defined by one of the following formulas:

$$\beta_k^{(HS)} = \frac{y_k^T g_{k+1}}{y_k^T d_k}; \; \beta_k^{(FR)} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}; \; \beta_k^{(PRP)} = \frac{y_k^T g_{k+1}}{g_k^T g_k} \quad \ldots\ldots\ldots\ldots\ldots (6)$$

$$\beta_k^{(CD)} = -\frac{g_{k+1}^T g_{k+1}}{g_k^T d_k}; \; \beta_k^{(LS)} = -\frac{y_k^T g_{k+1}}{g_k^T d_k}; \; \beta_k^{(DY)} = \frac{g_{k+1}^T g_{k+1}}{y_k^T s_k} \quad \ldots \; \ldots\ldots\ldots\ldots\ldots (7)$$

Al-Bayati and Al-Assady In (Al-Bayati and Al-Assady ,1986) proposed three forms for the scalar β_k defined by :

$$\beta_k^{AB1} = \frac{\|y_k\|^2}{\|g_k\|^2}; \beta_k^{AB2} = -\frac{\|y_k\|^2}{d_k^T g_k}; \beta_k^{AB3} = \frac{\|y_k\|^2}{d_k^T y_k}; \quad \ldots\ldots\ldots\ldots (8)\ [8]$$

3 PROPOSED A NEW CONJUGANCY COEFFICIENT

We have the quasi-Newton condition

$$y_k = G_k s_k \qquad (9)$$

We multiply both sides of equation (9) by y_k and we get

$$[y_k = G_k s_k] * s_k \Rightarrow y_k^T s_k = G s_k^T s_k$$

$$G = \frac{y_k^T s_k}{\|s_k\|^2} . I_{nxn} \qquad (10)$$

Let

$$d_{k+1}^N = -\lambda G_k^{-1} g_{k+1} \qquad (11)$$

$$d_{k+1}^N = -\lambda \frac{y_k^T s_k}{\|s_k\|^2} g_{k+1} \qquad (12)$$

Multiply both sides of equation (12) by y_k and we get

$$y_k^T d_{k+1}^N = -\lambda \left[\frac{y_k^T s_k}{\|s_k\|^2} \right] y_k^T g_{k+1} \qquad (13)$$

$$\Rightarrow y_k^T d_{k+1}^{CG} = -y_k^T g_{k+1} + \beta_k d_k^T y_k \qquad (14)$$

From (13) and (14) we have

$$-y_k^T g_{k+1} + \beta_k d_k^T y_k = -\lambda \left[\frac{y_k^T s_k}{\|s_k\|^2} \right] y_k^T g_{k+1}; \qquad (15)$$

We assume that

$$\beta_k = \beta_k^{(DY)} = \frac{g_{k+1}^T g_{k+1}}{y_k^T d_k}$$

Then we have

$$-y_k^T g_{k+1} + \beta_k^{DY} d_k^T y_k = -\lambda \left[\frac{y_k^T s_k}{\|s_k\|^2} \right] y_k^T g_{k+1}; \qquad (16)$$

$$-y_k^T g_{k+1} + \frac{\|g_{k+1}\|^2}{d_k^T y_k} d_k^T y_k = -\lambda \left[\frac{y_k^T s_k}{\|s_k\|^2} \right] y_k^T g_{k+1}; \qquad (17)$$

From eq.(17) we get:

$$-y_k^T g_{k+1} + \beta_k \tau_k = -\lambda \left[\frac{y_k^T s_k}{\|s_k\|^2} \right] y_k^T g_{k+1}; \qquad (18)$$

Then, we have

$$\beta_k = \frac{-\left[\frac{\|s_k\|^2}{2[f_k - f_{k+1} + g_{k+1}^T s_k]} \right] \left[\frac{y_k^T s_k}{\|s_k\|^2} \right] y_k^T g_{k+1} + y_k^T g_{k+1}}{\tau_k} \quad \ldots\ldots \; (19)$$

$$\beta_k = \frac{-\left[\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)} \right] y_k^T g_{k+1} + y_k^T g_{k+1}}{\tau_k} \quad \ldots. (20)$$

$$\beta_k = \frac{\left[1-\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\tau_k}, \quad (21)$$

Since τ_{k+1} then we suppose: $\tau_k = \|g_k\|^2$ then:

$$\beta_k = \frac{\left[1-\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2}; \quad (22)$$

4 OUTLINES OF THE PROPOSED ALGORITHM

Step(1):The initial step: We select starting point $x_0 \in R^n$, and we select the accuracy solution ε>0 is a small positive real number and we find $d_k=-g_k$, $\lambda_0=\text{argmin } \|g_0\|^2$, and we set k=0.

Step(2): The convergence test: If $\|g_0\| \le \varepsilon$ then stop and set the optimal solution is x_k. Else, go to step(3).

Step(3): The line search: We compute the value of λ_k by Cubic method and that satisfy the Wolfe conditions in Eqs.(4),(5) and go to step(4).

Step(4): Update the variables: $x_{k+1} = x_k + \lambda_k d_k$ and compute $f(x_{k+1}), g_{k+1}$. And $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step(5): Check: if $\|g_{k+1}\| \le \varepsilon$ then stop. Else continue.

Step (6): The search direction: We compute the scalar $\beta_k^{(New)}$ by use the equation (20) and set k=k+1, and go to step (4).

5 THE CONVERGENCE ANALYSIS

Theoretical Properties for the New CG-Method.

In this section, we focus on the convergence behavior on the β_k^{New} method with exact line searches. Hence, we make the following basic assumptions on the objective function.

Assumption(1):

f is bounded below in the level set $L_{x_0} = \{x \in R^n | f(x) \le f(x_0)\}$; in some neighborhood U of the level set L_{x0}, f is continuously differentiable and its gradient ∇f is Lipschitz continuous in the level set L_{x0}, namely, there exists a constant L> 0 such that:

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|; \ \text{for all } x, y \in L_{x0} \quad (23)$$

5.1 Sufficient Descent Property

We will show that in this section the proposed algorithm which defined in the equations (22) and (3) satisfy the sufficient descent property which satisfy the convergence property.

Theorem (1):

The search direction d_k that generated by the proposed algorithm of modified CG satisfy the descent property for all k, when the step size λ_k satisfied the Wolfe conditions (4),(5).

Proof: we will use the indication to prove the descent property, for k=0, $d_0 = -g_0 \Rightarrow d_0^T g_0 = -\|g_0\| < 0$, then we proved that the theorem is true for k=0,we assume that

$$\|s_k\| \le \eta; \ \|g_{k+1}\| \le \Gamma \ and \ \|g_k\| \le \eta 2$$

and assume that the theorem is true for any k, i.e. $d_k^T g_k < 0 \quad or \quad s_k^T g_k < 0 \quad or \ since \quad s_k = \lambda_k d_k$;

now we will prove that the theorem is true for k+1 then:

$$d_{k+1} = -g_{k+1} + \beta_k^{(New)} d_k \quad (24)$$

i.e.

$$d_{k+1} = -g_{k+1} + \frac{\left[1-\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2} d_k \quad \ldots\ldots\ldots\ldots\ldots (25)$$

Multiply both sides of the equation (25) by g_{k+1} we get:

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + \frac{\left[1-\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2} g_{k+1}^T d_k \quad \ldots\ldots\ldots\ldots\ldots (26)$$

$$g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2 \ge \frac{\left(1-\frac{y_k^T s_k}{\|y_k\|^2}\right) y_k^T g_{k+1}}{\lambda\|g_k\|^2 + (1-\lambda)\|d_k\|\|y_k\|} g_{k+1}^T d_k \quad (27)$$

Divided both side by $\|g_{k+1}\|^2$:

$$\left[\frac{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2}{\|g_{k+1}\|^2} = \frac{\left[1-\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2} \frac{g_{k+1}^T d_k}{\|g_{k+1}\|^2}\right] \quad (28)$$

$$\frac{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2}{\|g_{k+1}\|^2} \le \frac{\left[1 - \frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] \|y_k\| \|g_{k+1}\|}{\|g_k\|^2} \frac{\|g_{k+1}\| \|d_k\|}{\|g_{k+1}\|^2}$$

$$\frac{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2}{\|g_{k+1}\|^2} \le \frac{\left[1 - \frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] \|y_k\| \|d_k\|}{\|g_k\|^2}; \quad (30)$$

$$\frac{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2}{\|g_{k+1}\|^2} \le \frac{\left[1 - \frac{\|y_k\| \|s_k\|}{2(f_k - f_{k+1} + \|g_{k+1}\| \|s_k\|)}\right] \|y_k\| \|d_k\|}{\|g_k\|^2}; \quad (31)$$

$$\frac{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2}{\|g_{k+1}\|^2} \le \frac{\|y_k\| \|d_k\|}{\|g_k\|^2} \quad (32)$$

$$\frac{\|g_{k+1}\|^2}{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2} \ge \frac{\|g_k\|^2}{\|y_k\| \|d_k\|} = \delta > 1; \quad (33)$$

$$\frac{g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2}{\|g_{k+1}\|^2} \le \frac{1}{\delta} \quad (34)$$

$$g_{k+1}^T d_{k+1} + \|g_{k+1}\|^2 \le \frac{1}{\delta} \|g_{k+1}\|^2; \quad (35)$$

$$g_{k+1}^T d_{k+1} \le -(1 - \frac{1}{\delta}) \|g_{k+1}\|^2$$

Let

$$c = (1 - \frac{1}{\delta}); \quad (36)$$

Then

$$g_{k+1}^T d_{k+1} \le -c \|g_{k+1}\|^2 \quad (37)$$

For some positive constant c>0. This condition has often been used to analyze the global convergence of conjugate gradient methods with inexact line search.

5.2 Global Convergence Property

The conclusion of the following lemma is used to prove the global convergence of nonlinear conjugate gradient methods, under the general Wolfe line search.

Lemma 1:

Suppose assumptions (1) (i) and (ii) hold and consider any conjugate gradient method (22) and (3), where dk is a descent direction and λk is obtained by the strong Wolfe line search. If

$$\sum_{k \ge 1}^{\alpha} \frac{1}{\|d_k\|^2} = \alpha \quad (38)$$

Then

$$\liminf_{k \to \infty} \|g_k\| = 0 \quad (39)$$

For uniformly convex functions which satisfy the above assumptions, we can prove that the norm of dk+1 given by (25) is bounded above. Assume that the function f is a uniformly convex function, i.e. there exists a constant $\mu \ge 0$ such that for all x,y ∈S,

$$(g(x) - g(y))^T (x - y) \ge \mu \|x - y\|^2, \quad (40)$$

Using lemma 1 the following result can be proved.

Theorem 2:

Suppose that the assumptions (i) and (ii) hold. Consider the algorithm (3), (22). If $\|s_k\|$ tends to zero and there exists nonnegative constants $\eta 1$ and $\eta 2$ such that:

$$\|g_k\|^2 \ge \eta 1 \|s_k\|^2; \quad \|g_{k+1}\|^2 \ge \eta 2 \|s_k\| \quad (41)$$

and f is a uniformly convex function, then.

$$\liminf_{k \to \infty} \|g_k\| = 0; \quad (42)$$

Proof: From eq. (22) We have:

$$\beta_k^{new} = \frac{\left[1 - \frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2}$$

From Cuchy-Shwartz we get:

$$|\beta_{k+1}^{New}| = \left| \frac{\left[1 - \frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2} \right| \le \frac{\left[1 - \frac{\|y_k\| \|s_k\|}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] \|y_k\| \|g_{k+1}\|}{\|g_k\|^2}; \quad (43)$$

But

$$\|y_k\| \le L\|s_k\|,.$$

Then

$$\left|\beta_{k+1}^{New}\right| = \left|\frac{\left[1-\frac{y_k^T s_k}{2(f_k - f_{k+1} + g_{k+1}^T s_k)}\right] y_k^T g_{k+1}}{\|g_k\|^2}\right| \le \frac{\left[1-\frac{L\|s_k\|\|s_k\|}{2(f_k - f_{k+1} + \|g_{k+1}\|\|s_k\|)}\right] L\|s_k\|\|g_{k+1}\|}{\|g_k\|^2}$$

………………. (44)

$$\left|\beta_{k+1}^{New}\right| \le \frac{\left[1-\frac{L\|s_k\|\|s_k\|}{2(f_k - f_{k+1} + \|g_{k+1}\|\|s_k\|)}\right] L\|s_k\|\|g_{k+1}\|}{\|g_k\|^2}$$

…………………….. (45)

From (41)

$$\left|\beta_{k+1}^{New}\right| \le \frac{\left[1-\frac{L\eta^2}{2(|f_k - f_{k+1}| + \eta\Gamma)}\right] L\eta\Gamma}{\eta 1\eta \|s_k\|} \qquad (46)$$

Let from theorem (1):

$$A = (f_k - f_{k+1})$$

then

$$\left|\beta_{k+1}^{New}\right| \le \frac{\left[1-\frac{L\eta^2}{2(|A| + \eta\Gamma)}\right] L\eta\Gamma}{\eta 1\eta \|s_k\|} \qquad (47)$$

$$\left|\beta_{k+1}^{New}\right| \le \frac{L\eta\Gamma}{\eta 1\eta \|s_k\|} \qquad (48)$$

Hence,

$$\|d_{k+1}\| \le \|g_{k+1}\| + \left|\beta_k^N\right| \|s_k\|; \qquad (49)$$

$$\|d_{k+1}\| \le \gamma + \frac{L\eta\Gamma}{\eta 1\eta \|s_k\|}\|s_k\| = \gamma + \frac{L\eta\Gamma}{\eta 1\eta} \qquad (50)$$

$$\sum_{k\ge 1} \frac{1}{\|d_{k+1}\|^2} = \infty \qquad (51)$$

$$\frac{1}{\left(\gamma + \frac{L\eta\Gamma}{\eta 1\eta}\right)^2} \sum_{k\ge 1} 1 = \infty \qquad (52)$$

6 DRAGONFLY ALGORITHM

Dragonflies are one of the types of flying insects, which may reach about 3000 species, and dragonflies are predators so some of them called the devil needle or the devil's arrow accordingly [4].

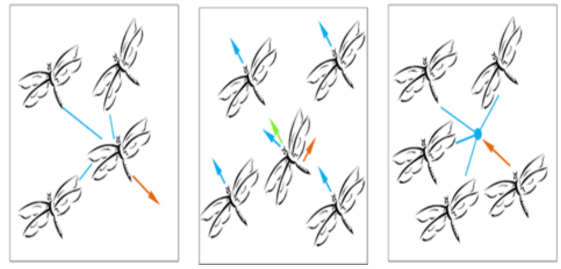
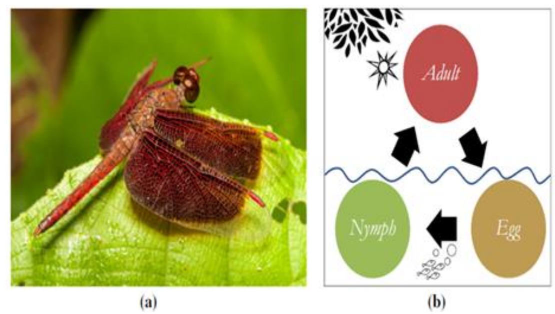


Fig. 1. a) the shape of a true dragonfly insect



b) the life cycle of dragonflies

6.1 *Dragonfly Algorithm*

Swarm behavior follows three basic principles of exploration and exploitation:

Separation: This refers to the constant avoiding collision of individuals with other individuals in the neighborhood.

• Alignment: that indicates matching the speed of individuals with other individuals in the neighborhood.

Cohesion: which indicates the tendency of individuals towards the center of the neighborhood block. As shown in the numbered Fig. (2

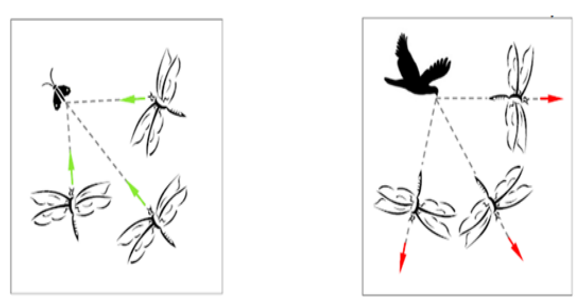


Fig. 2. Primitive Correction Patterns Among Swarm Individuals

attraction towards a food source Leaving away from the enemies

he above behaviors are modeled mathematically as follows:

The principle of separation is calculated as follows:

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N} \quad (53)$$

whereas :

Vj: represents the velocity of j of the adjacent individuals.

The principle of cohesion is calculated mathematically as follows:

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X \quad (54)$$

As: X: represents the current individual location, Xj: represents the j location of the adjacent individuals, and N: the number of adjacent individuals.

The principle of attraction to a food source is calculated as follows:

$$F_i = X^+ - X \quad (55)$$

Whereas:

X: represents the individual's current location and X +: represents the location of the food source.

Finally, the principle of leaving away from enemies is calculated as follows:

$$E_i = X^- + X \quad (56)$$

As: X: represents the individual's current location and X- : represents the enemy's location.

It is assumed that the dragonflies behavior is a combination of these five corrective patterns. To update the position of the artificial dragonflies in the research area and simulate their movements, two vectors are taken into consideration, namely: step (ΔX) and location (X). The DA algorithm has been developed based on the PSO algorithm. The step vector shows the direction of motion of the dragonfly (note that the site-update model of the artificial dragonfly is defined in one dimension, but the proposed method can extend to higher dimensions) and is defined as follows:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + w\Delta X_t \quad \ldots\ldots\ldots\ldots\ldots \quad (57)$$

As: (s) represents the weight of the separation and (Si) indicates the separation for (i) of the individuals, (a) is the weight of the alignment, (A) is the alignment for (i) of the individuals, and (c) indicates the weight of the cohesion, (Ci) is the coherence for (i) of the individual, and (f) is the food factor, (Fi) is the source food for (i) of an individual, (e) is the enemy factor, (Ei) is the enemy location for (i) of an individual ,(w) is the initial weight, and (t) is the repeater counter.

After calculating the step vector, the location vector is calculated as follows:

$$X_{t+1} = X_t + \Delta X_{t+1} \quad (58)$$

As: t is the current iteration.

To improve randomness, random behavior, and exploration of artificial dragonflies, the dragonflies swarm is required to fly around the search space using the (Levy flight) method when there are no adjacent solutions. In this case, the dragonfly's site is updated with the following formula:

$$X_{t+1} = X_t + \mathrm{Levy}(d) * X_i \quad (59)$$

Where: t is the current iteration, d: is the dimension of the location vector. The (Levy flight) equation is calculated as follows:

$$\mathrm{Levy}(x) = 0.01 * \frac{r_1 * \sigma}{|r_2|^{\frac{1}{\beta}}}$$

(60)

Where: r1, r2 are random numbers enclosed between [0,1], β: constant (equal to 1.5) and that σ is calculated as follows:

$$\sigma = \left(\frac{r(1+\beta)*\sin(\frac{\pi\beta}{2})}{F\left(\frac{1+\beta}{2}\right)*\beta*2^{(\frac{\beta-1}{2})}}\right)^{\frac{1}{\beta}} \quad (61)$$

WHERE : $F(x) = (x-1)!$

6.2 The Steps of the Dragonfly Algorithm

The steps for the Dragonfly DA algorithm can be summarized in below:

Step (1): Configure the dragon community Xi (i = 1, 2, ..., n).

Step (2): Initialize the vector of step ΔXi (i = 1, 2, .. ., n)

Step (3): When the stopping condition is not met (access to max- iter.).

Step (4): Calculate the target function value for all dragonflies.

Step (5): Update the source of the food and the enemy according to Eqs.(4), (5).

Step (6): Update the values for (w, s, a, c, f, e).

Step (7): Calculate the values of (S, A, C, F, E) using equations 1 to 5.

Step (8): Update the Radius beam to the Neighborhood.

Step (9): If the dragonfly has at least one of the neighboring dragonflies, then update the velocity vector using equation (57) and the location vector using equation (58) otherwise go to step (10).

Step (10): Otherwise, update the location vector using equation (49).

Step (11): Verify and correct new locations based on variable limits and finish.

7 PROPOSED HYBRID ALGORITHM

In this section, a new hybrid method has been proposed to solve the optimization as in the following flow chart :

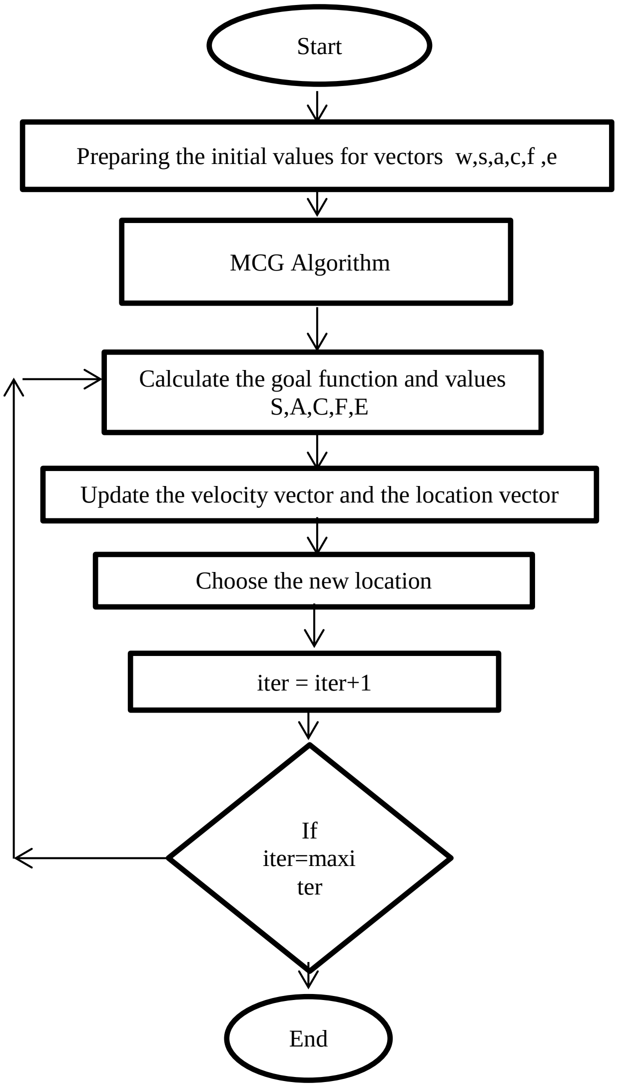


Fig. 3. Flowchart Of The Proposed Algorithm (Da-Mcg)

That have been called DA-MCG A proposed hybrid Algorithm, called DA- MCG. The steps of the proposed hybrid algorithm (DA-MCG)

8 NUMERICAL RESULTS

For the purpose of evaluating the performance of the proposed algorithms in solving optimization issues, the proposed algorithm was tested DA-MCG, using (10) standard functions in order to compare with the dragonflies algorithm itself. Table (1) shows the details of the test functions. The stopping condition is used if the function reaches the minimum value and the highest frequency of all programs is equal to (500) repetitions.

Function	Dim	Range	Fmin
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-100,100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,10]	0
$F_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{n} x_i)^2$	30	[-100,100]	0
$F_4(x) = \max_i \{\lvert x_i \rvert . 1 \le i \le n\}$	30	[-100,100]	0
$F_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,30]	0
$F_6(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12,5.12]	0
$F7(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum}\right. + 20 + e$	30	[-32,32]	0
$F_8(x) = \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600,600]	0
$F_9(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[-5,5]	0.00030
$F_{10} = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1.0316

As for the numbered tables (2,3,4), it shows the results of the algorithm (DA-MCG) compared to the results of the algorithm (DA), as it shows the success of the proposed algorithm (DA-MCG) by improving the results of most of the standard high-performance test functions and this confirms the success of Hybridization process.

Table 2: Comparison the results between DA and DA-MCG using the number of elements consisting of 15 elements and number of iterations 500

Func.	DA	DA-MCG
F_1	0.0422	2.88401300000 e-198
F_2	0.0093	4.0600000000 e-100
F_3	0.0851	3.357760000e-198
F_4	0.0260	1.71242000000e-99
F_5	4.2346	3
F_6	2.2909	0
F_7	0.3821	8.8818 e-16
F_8	0.2228	0
F_9	0.0043	0.14841
F_{10}	-1.0316	-1.6494000278 e-52

Table 3: Comparison the results between DA and DA-MCG using the number of elements consisting of 20 elements and number of iterations 500

Func.	DA	DA-MCG
F_1	3.6151e-05	6.868900000e-198
F_2	2.0277e-08	3.2190710000 e-100
F_3	0.18831746555	3.336236666 e-198
F_4	1.8951e-05	1.98945400000e-99
F_5	3.55604000000	3
F_6	4.08238090909	0
F_7	3.680815 e-08	8.8818 e-16
F_8	0.3106550	0
F_9	0.00594456875	0.14841
F_{10}	-1.0316	-1.5265e-103

Table 4: Comparison the results between DA and DA-MCG using the number of elements consisting of 30 elements and number of iterations 500

Func.	DA	DA-MCG
F_1	7.9415 e-06	2.583248000000000 e-198
F_2	3.5461 e-06	2.647690000000000 e-100
F_3	5.4455 e-07	2.566200000000000 e-198
F_4	4.67 e-06	1.164059000000000 e-99
F_5	59.4639409 09	3
F_6	1.9899	0
F_7	2.552 e-08	8.8818 e-16
F_8	0	0
F_9	0.0016	0.14841
F_{10}	-1.0316	-2.122295805409 e-43

The test was applied by a laptop that carries the following characteristics: the processor speed is 2.70, the memory size is 8GB, and the Matlab R2014a program is running Windows 8.

9 CONCLUSIONS

Hybridization of heuristic algorithms with one of the modified classical algorithms contributed to improving its performance by increasing the speed of convergence, and also led to an improvement in the quality of the resulting solutions by increasing its exploratory and exploitative capabilities, as numerical results showed the ability of hybrid algorithms to solve various optimization issues. The results of the DA- MCG algorithm were compared with the algorithm of examples of dragonflies themselves, the DA, which resulted in encouraging results.

10 REFERENCES

[1] J. Nocedal and S. J. Wright, "Numerical optimization 2nd," Springer2006.
[2] P. Moallem, S. A. Monadjemi, B. Mirzaeian and M. Ashourian, A novel fast backpropagation learning algorithm using parallel tangent and heuristic line search, Proceedings of the 10th WSEAS international conference on Computers,
[3] World Scientific and Engineering Academy and Society (WSEAS), 2006, pp. 634-639.
[4] Meng, X., Liu, Y., Gao, X., & Zhang, H. (2014). A new bio-inspired algorithm: chicken swarm optimization. International conference in swarm intelligence, pp.86-94

[5] Reddy, P. D. P., Reddy, V. V., & Manohar, T. G. (2017). Whale optimization algorithm for optimal sizing of renewable resources for loss reduction in distribution systems. Renewables: Wind, Water, and Solar, Vol.4(1), pp.1-13.

[6] Bashishtha, T. K., & Srivastava, L. (2016). Nature inspired meta- heuristic dragonfly algorithms for solving optimal power flow problem. International Journal of Electronics, Electrical and Computational System, Vol.5(5),pp.111-120.

[7] Pathania, A. K., Mehta, S., & Rza, C. (2016). Multi-objective dispatch of thermal system using dragonfly algorithm International Journal of Engineering Research, Vol.5(11), pp. 861-866.

[8] AL – Bayati, A.Y. and AL – Assady, N.H. (1986). " Conjugate gradient method", Technical Research , school of computer studies , Leeds University.