

Parameter-Free Deterministic Global Search with Simplified Central Force Optimization

Richard A. Formato

Registered Patent Attorney & Consulting Engineer
P.O. Box 1714, Harwich, MA 02645, USA
rf2@ieee.org

Abstract. This note describes a simplified parameter-free implementation of Central Force Optimization for use in deterministic multidimensional search and optimization. The user supplies only the objective function to be maximized, nothing more. The algorithm's performance is tested against a widely used suite of twenty three benchmark functions and compared to other state-of-the-art algorithms. CFO performs very well.

Keywords: Central Force Optimization, CFO, Deterministic Algorithm, Multi-dimensional Search and Optimization, Parameter-Free Optimization, Gravitational Kinematics Metaphor, Metaheuristic.

1 Introduction

Central Force Optimization (CFO) is a *deterministic* Nature-inspired metaheuristic for an evolutionary algorithm (EA) that performs multidimensional global search and optimization [1-13]. This note presents a simplified parameter-free CFO implementation that requires only one user-specified input: the objective function to be maximized. The algorithm performs quite well across a wide range of functions.

A major frustration with many EAs is the plethora of setup parameters. For example, the three Ant Colony Optimization algorithms described in [14] (AS, *MAX-MIN*, ACS) require the user to specify nine parameters. The generalized Particle Swarm Optimization algorithm in [15] requires the user to specify a population size and six "suitable bounded coefficients". This frustration is compounded by the facts that (a) there usually is no methodology for picking "good" values; (b) the "right" parameters are often problem-specific; (c) the solutions often are sensitive to small changes; and (d) run upon run, exactly the same parameters never yield the same results because the algorithm is *inherently stochastic*.

CFO is quite different. It is based on Newton's laws of motion and gravity for real masses moving through the real Universe; and just as these laws are mathematically precise, so too is CFO. It is completely deterministic at every step, with successive runs employing the same setup parameters yielding precisely the same results. Its inherent determinism distinguishes CFO from all the stochastic EAs that fail completely if randomness is removed. Moreover, CFO's metaphor of gravitational kinematics appears to be more than a simple analogy. While many metaheuristics are

inspired by natural processes, ant foraging or fish swarming, for example, the resulting algorithm is not (necessarily) an accurate mathematical model of the actual process. Rather, it truly is a metaphor. By contrast, CFO and real gravity appear to be much more closely related. Indeed, CFO's "metaphor" actually may be reality, because its "probes" often exhibit behavior that is strikingly similar to that of gravitationally trapped near earth objects (NEOs). If so, the panoply of mathematical tools and techniques available in celestial mechanics may be applicable to CFO directly or with modification or extension (see [1,7-9]). CFO may be interpreted in different ways depending upon the underlying model (vector force field, gravitational potential, kinetic or total energy) [3,13]. These observations, although peripheral to the theme of this note, provide additional context for the CFO metaheuristic.

There are, of course, other gravity-inspired metaheuristics, notably Space Gravitational Optimization [16]; Integrated Radiation Optimization [17]; and Gravitational Search Algorithm [18, 19]. But each one is *inherently* stochastic. Their basic equations contain true random variables whose values must be computed from a probability distribution and consequently are unknowable in advance. CFO, by contrast, is completely deterministic. Even arbitrarily assigned variables (for example, the "repositioning factor" discussed below), are known with absolute precision. At no point in a CFO run is any value computed probabilistically, which is a major departure from how almost all other Nature-inspired metaheuristics function.

2 CFO Algorithm

Pseudocode for the parameter-free CFO implementation appears in Fig. 1. Simplification of the algorithm is accomplished by hardwiring all of CFO's basic parameters, which results in simplified equations of motion as described below. CFO searches for the global *maxima* of an *objective function* $f(x_1, x_2, \dots, x_{N_d})$ defined on the N_d -dimensional decision space $\Omega: x_i^{\min} \leq x_i \leq x_i^{\max}, 1 \leq i \leq N_d$. The x_i are *decision variables*, and i the coordinate number. The value of $f(\vec{x})$ at point \vec{x} in Ω is its *fitness*. $f(\vec{x})$'s topology or "landscape" in the N_d -dimensional hyper-space is unknown, that is, there is no *a priori* information about the objective function's maxima. CFO searches Ω by flying "probes" through the space at discrete "time" steps (iterations). Each probe's location is specified by its position vector computed from two *equations of motion* that analogize their real-world counterparts for material objects moving through physical space under the influence of gravity without energy dissipation.

Probe p 's position vector at step j is $\vec{R}_j^p = \sum_{k=1}^{N_d} x_k^{p,j} \hat{e}_k$, where the $x_k^{p,j}$ are its coordinates and \hat{e}_k the unit vector along the x_k -axis. The indices j , $0 \leq j \leq N_t$, and p , $1 \leq p \leq N_p$, respectively, are the iteration number and probe number, with N_t and N_p being the corresponding *total* numbers of time steps and probes.

In metaphorical “CFO space” each of the N_p probes experiences an acceleration created by the “gravitational pull” of “masses” in Ω (see [1,12] for details). Probe p ’s acceleration at step $j-1$ is given by

$$\vec{a}_{j-1}^p = \sum_{\substack{k=1 \\ k \neq p}}^{N_p} U(M_{j-1}^k - M_{j-1}^p) \cdot (M_{j-1}^k - M_{j-1}^p) \times \frac{(\vec{R}_{j-1}^k - \vec{R}_{j-1}^p)}{\|\vec{R}_{j-1}^k - \vec{R}_{j-1}^p\|}, \quad (1)$$

which is the first of CFO’s two equations of motion. In (1), $M_{j-1}^p = f(x_1^{p,j-1}, x_2^{p,j-1}, \dots, x_{N_d}^{p,j-1})$ is the objective function’s fitness at probe p ’s location at time step $j-1$. Each of the other probes at that step (iteration) has associated with it fitness $M_{j-1}^k, k = 1, \dots, p-1, p+1, \dots, N_p$. $U(\cdot)$ is the

Unit Step function, $U(z) = \begin{cases} 1, & z \geq 0 \\ 0, & \text{otherwise} \end{cases}$. If $\vec{R}_{j-1}^k = \vec{R}_{j-1}^p, k \neq p$, then

$\vec{a}_{j-1}^p \equiv 0$ because $M_{j-1}^k = M_{j-1}^p$ and \vec{a}_{j-1}^p is indeterminate (probes k and p have coalesced so that k cannot exert a gravitational force on p). The acceleration \vec{a}_{j-1}^p causes probe p to move from position \vec{R}_{j-1}^p at step $j-1$ to position \vec{R}_j^p at step j according to the trajectory equation

$$\vec{R}_j^p = \vec{R}_{j-1}^p + \vec{a}_{j-1}^p, \quad j \geq 1, \quad (2)$$

which is CFO’s second equation of motion. These simplified CFO equations result from hardwiring CFO’s basic parameters to $G=2, \Delta t=1, \alpha=1$, and $\beta=1$ (see [1,11,12] for definitions and discussion). The “internal parameters” in Fig. 1 are *not* fundamental CFO parameters, but rather specific to this *particular* implementation.

Every CFO run begins with an Initial Probe Distribution (IPD) defined by two variables: (a) the total number of probes used, N_p ; and (b) where the probes are placed inside Ω . The IPD used here is an orthogonal array of N_p / N_d probes per dimension deployed uniformly on “probe lines” parallel to the coordinate axes and intersecting at a point that slides along Ω ’s principal diagonal. Fig. 2 provides a two-dimensional (2D) example of this type of IPD (9 probes shown on each probe line, 2 overlapping, but any number may be used). The probe lines are parallel to the x_1 and x_2 axes intersecting at a point on Ω ’s principal diagonal marked by position vector $\vec{D} = \vec{X}_{\min} + \gamma(\vec{X}_{\max} - \vec{X}_{\min})$. The diagonal’s endpoints are at

Procedure $CFO[f(\vec{x}), N_d, \Omega]$

Internals: N_t , F_{rep}^{init} , ΔF_{rep} , F_{rep}^{min} , $\left(\frac{N_p}{N_d}\right)_{MAX}$, γ_{start} , γ_{stop} , $\Delta\gamma$.

Initialize $f_{max}^{global}(\vec{x}) = \text{very large negative number, say, } -10^{+4200}$.

For $N_p/N_d = 2$ to $\left(\frac{N_p}{N_d}\right)_{MAX}$ by 2:

(a.0) Total number of probes: $N_p = N_d \cdot \left(\frac{N_p}{N_d}\right)$

For $\gamma = \gamma_{start}$ to γ_{stop} by $\Delta\gamma$:

(a.1) Re-initialize data structures for position/acceleration vectors & fitness matrix.

(a.2) Compute IPD [see text].

(a.3) Compute initial fitness matrix, $M_0^p, 1 \leq p \leq N_p$.

(a.4) Initialize $F_{rep} = F_{rep}^{init}$.

For $j=0$ to N_t [or earlier termination - see text]:

(b) Compute position vectors, $\vec{R}_j^p, 1 \leq p \leq N_p$ [eq. (2)].

(c) Retrieve errant probes ($1 \leq p \leq N_p$):

If $\vec{R}_j^p \cdot \hat{e}_i < x_i^{\min} \therefore \vec{R}_j^p \cdot \hat{e}_i = \max\{x_i^{\min} + F_{rep}(\vec{R}_{j-1}^p \cdot \hat{e}_i - x_i^{\min}), x_i^{\min}\}$.

If $\vec{R}_j^p \cdot \hat{e}_i > x_i^{\max} \therefore \vec{R}_j^p \cdot \hat{e}_i = \min\{x_i^{\max} - F_{rep}(x_i^{\max} - \vec{R}_{j-1}^p \cdot \hat{e}_i), x_i^{\max}\}$.

(d) Compute fitness matrix for current probe distribution, $M_j^p, 1 \leq p \leq N_p$.

(e) Compute accelerations using current probe distribution and fitnesses [eq. (1)].

(f) Increment F_{rep} : $F_{rep} = F_{rep} + \Delta F_{rep}$; If $F_{rep} > 1 \therefore F_{rep} = F_{rep}^{min}$.

(g) If $j \geq 20$ and $j \text{ MOD } 10 = 0 \therefore$

(i) Shrink Ω around \vec{R}_{best} [see text].

(ii) Retrieve errant probes [procedure Step (c)].

Next j

(h) Reset Ω boundaries [values before shrinking].

(i) If $f_{max}(\vec{x}) \geq f_{max}^{global}(\vec{x}) \therefore f_{max}^{global}(\vec{x}) = f_{max}(\vec{x})$.

Next γ

Next N_p/N_d

Fig. 1. Parameter-free CFO Pseudocode

$\vec{X}_{\min} = \sum_{i=1}^{N_d} x_i^{\min} \hat{e}_i$ and $\vec{X}_{\max} = \sum_{i=1}^{N_d} x_i^{\max} \hat{e}_i$, and parameter $0 \leq \gamma \leq 1$ determines where along the diagonal the probe lines intersect.

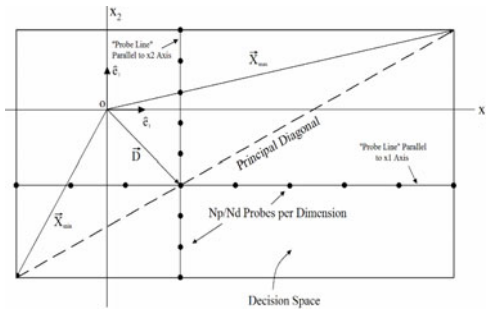


Fig. 2. Variable 2D Initial Probe Distribution

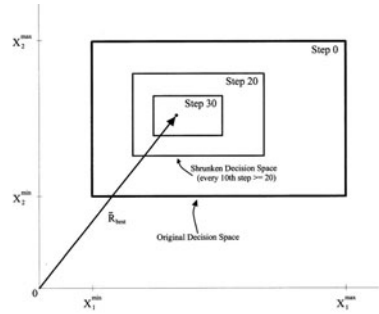


Fig. 5. 2D Decision Space Adaptation

Fig. 3 shows a typical 2D IPD for different values of γ , while Fig. 4 provides a 3D example. In Fig. 4 each probe line contains 6 equally spaced probes. For certain values of γ three probes overlap at the probe lines’ intersection point on the principal diagonal. Of course, this IPD procedure is generalized to the N_d -dimensional decision space Ω to create N_d probe lines parallel to the N_d coordinate axes. While Fig. 2 shows equal numbers of probes on each probe line, a different number of probes per dimension can be used instead. For example, if equal probe spacing were desired in a decision space with unequal boundaries, or if overlapping probes were to be excluded in a symmetrical space, then unequal numbers could be used. Unequal numbers might be appropriate if *a priori* knowledge of Ω ’s landscape, however obtained, suggests denser sampling in one region.

Errant probes are a concern in CFO because a probe’s acceleration computed from equation (1) may be too great to keep it inside Ω . If any coordinate $x_i < x_i^{\min}$ or $x_i > x_i^{\max}$, the probe enters a region of *unfeasible* solutions that are not valid for the problem at hand. The question (which arises in many algorithms) is what to do with an errant probe. While many schemes are possible, a simple, empirically determined one is used here. On a coordinate-by-coordinate basis, probes flying outside Ω are placed a fraction $F_{rep}^{\min} \leq F_{rep} \leq 1$ of the distance between the probe’s starting coordinate and the corresponding boundary coordinate. F_{rep} is the “repositioning factor” introduced in [2]. See step (c) in the pseudocode of Fig. 1 for details. F_{rep} starts at an arbitrary initial value F_{rep}^{init} which is then incremented by an arbitrary amount ΔF_{rep}

at each iteration (subject to $F_{rep}^{min} \leq F_{rep} \leq 1$). This procedure provides better sampling of Ω by distributing probes throughout the space.

This particular CFO implementation includes adaptive reconfiguration of Ω to improve convergence speed. Fig. 5 illustrates in 2D how Ω 's size is adaptively reduced around \vec{R}_{best} , the location of the probe with best fitness throughout the run up to the current iteration. Ω shrinks every 10th step beginning at step 20. Its boundary coordinates are reduced by one-half the distance from the best probe's position to each boundary on a coordinate-by-coordinate basis, that is, $x_i'^{min} = x_i^{min} + \frac{\vec{R}_{best} \cdot \hat{e}_i - x_i^{min}}{2}$ and $x_i'^{max} = x_i^{max} - \frac{x_i^{max} - \vec{R}_{best} \cdot \hat{e}_i}{2}$, where the primed coordinate is Ω 's new boundary, and the dot denotes vector inner product. For clarity, Fig. 5 shows \vec{R}_{best} as being fixed, but generally it varies throughout a run. Changing Ω 's boundary every 10 steps instead of some other interval is arbitrary.

The internal parameter values were: $N_t = 1000$, $F_{rep}^{init} = 0.5$, $\Delta F_{rep} = 0.1$, $F_{rep}^{min} = 0.05$, $\left(\frac{N_p}{N_d}\right)_{MAX} = 14$ for $N_d \leq 6$, $\left(\frac{N_p}{N_d}\right)_{MAX} = 6$ for $21 \leq N_d \leq 30$, $\gamma_{start} = 0$, $\gamma_{stop} = 1$, $\Delta\gamma = 0.1$, with N_t reduced to 100 for function f_7 because

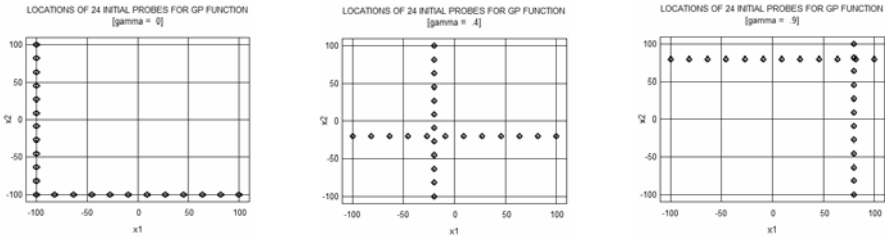


Fig. 3. Typical 2D IPD's for Different Values of γ

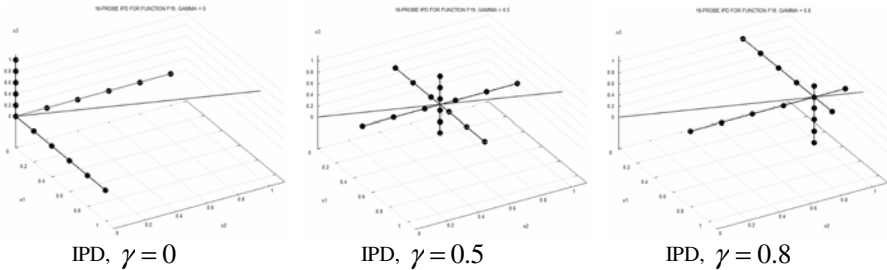


Fig. 4. Typical Variable 3D IPD for the GSO f_{19} Function (probes as filled circles)

this benchmark contains a random component that may result in excessive runtimes. The test for early termination is a difference between the average best fitness over 25 steps, including the current step, and the current best fitness of less than 10^{-6} (absolute value). This test is applied starting at iteration 35 (at least 10 steps must be computed before testing for saturation). The internal parameters, indeed all CFO-related parameters, were chosen empirically. These particular values provide good results across a wide range of test functions. Beyond this empirical observation there currently is no methodology for choosing either CFO's basic parameters or the internal parameters for an implementation such as this one. There likely are better parameter sets, which is an aspect of CFO development that merits further study.

3 Results

Group Search Optimizer (GSO) [20,21] is a new stochastic metaheuristic that mimics animal foraging using the strategies of “producing” (searching for food) and “scrounging” (joining resources). GSO was tested against 23 benchmark functions from 2 to 30D. GSO thus provides a superior standard for evaluating CFO. CFO therefore was tested against the same 23 function suite. The results are reported here.

In [20], GSO (inherently *stochastic*) was compared to two other stochastic algorithms, “PSO” and “GA”. PSO is a Particle Swarm implemented using “PSOt” (MATLAB toolbox of standard and variant algorithms). The standard PSO algorithm was used with recommended default parameters: population, 50; acceleration factors, 2.0; inertia weight decaying from 0.9 to 0.4. GA is a real-coded Genetic Algorithm implemented with GAOT (genetic algorithm optimization toolbox). GA had a fixed population size (50), uniform mutation, heuristic crossover, and normalized geometric ranking for selection with run parameters set to recommended defaults. A detailed discussion of the experimental method and the toolboxes is at [20, p. 977].

Thus, while this note compares CFO and GSO directly, it indirectly compares CFO to PSO and GA as well. Table 1 summarizes CFO's results. F is the test function using the same function numbering as [20]. f_{\max} is the known global *maximum* (note that the negative of each benchmark in [20] is used here because, unlike the other algorithms, CFO locates maxima, not minima). N_{eval} is the total number of function evaluations made by CFO. $\langle \cdot \rangle$ denotes average value. Because GSO, PSO and GA are inherently stochastic, their performance must be described statistically. The statistical data in Table 1 for those algorithms are reproduced from [20].

Of course, no statistical description is needed for CFO because CFO is inherently *deterministic* (but see [11] for a discussion of *pseudorandomness* in CFO). For a given setup, only one CFO run is required to assess its performance. In the first group of high dimensionality unimodal functions ($f_1 - f_7$), CFO returned the best fitness on all seven functions, in one case by a wide margin (f_5). In the second set of six high dimensionality multimodal functions with many local maxima ($f_8 - f_{13}$), CFO performed best on three ($f_9 - f_{11}$) and essentially the same as GSO on f_8 . In the last group of ten multimodal functions with few local maxima ($f_{14} - f_{23}$), CFO returned the best fitness on four ($f_{20} - f_{23}$, and by wide margins on $f_{21} - f_{23}$); equal fitnesses on four (f_{14} , f_{17} , f_{18} , f_{19}); and only very slightly lower fitness on two (f_{15} , f_{16}).

CFO did not perform quite as well as the other algorithms on only 2 of the 23 benchmarks (f_{12}, f_{13}). But its performance may be improved if a second run is made using a smaller Ω based on the first run. For example, for f_{12} the 30 coordinates returned on the run #1 are in the range $[-1.03941458 \leq x_i \leq -0.99688049]$ (known maximum at $[-1]^{30}$). If based on this result Ω is shrunk from $[-50 \leq x_i \leq 50]$ to $[-5 \leq x_i \leq 5]$ and CFO is run again, then on run #2 the best fitness improves to -7.39354×10^{-35} ($N_{eval} = 273,780$), which is a good bit better than GSO's result.

Table 1. CFO Comparative Results for 23 Benchmark Suite in [20]

(^{*} negative of the functions in [20] are computed by CFO because CFO searches for maxima instead of minima).

F^*	N_d	f_{max}^*	<Best Fitness>/ Other Algorithm	--- CFO ---	
				Best Fitness	N_{eval}
Unimodal Functions (other algorithms: average of 1000 runs)					
f_1	30	0	-3.6927x10 ⁻³⁷ / PSO	0	222,960
f_2	30	0	-2.9168x10 ⁻²⁴ / PSO	0	237,540
f_3	30	0	-1.1979x10 ⁻³ / PSO	-6.1861x10 ⁻⁵	397,320
f_4	30	0	-0.1078 / GSO	0	484,260
f_5	30	0	-37.3582 / PSO	-4.8623x10 ⁻⁵	436,680
f_6	30	0	-1.6000x10 ⁻² / GSO	0	176,580
f_7	30	0	-9.9024x10 ⁻³ / PSO	-1.2919x10 ⁻⁴	399,960
Multimodal Functions, Many Local Maxima (other algorithms: avg 1000 runs)					
f_8	30	12,569.5	12,569.4882 / GSO	12,569.4865	415,500
f_9	30	0	-0.6509 / GA	0	397,080
f_{10}	30	0	-2.6548x10 ⁻⁵ / GSO	4.7705x10 ⁻¹⁸	518,820
f_{11}	30	0	-3.0792x10 ⁻² / GSO	-1.7075x10 ⁻²	235,800
f_{12}	30	0	-2.7648x10 ⁻¹¹ / GSO	-2.1541x10 ⁻⁵	292,080
f_{13}	30	0	-4.6948x10 ⁻⁵ / GSO	-1.8293x10 ⁻³	360,000
Multimodal Functions, Few Local Maxima (other algorithms: avg 50 runs)					
f_{14}	2	-1	-0.9980 / GSO	-0.9980	78,176
f_{15}	4	-3.075x10 ⁻⁴	-3.7713x10 ⁻⁴ / GSO	-5.6967x10 ⁻⁴	143,152
f_{16}	2	1.0316285	1.031628 / GSO	1.03158	87,240
f_{17}	2	-0.398	-0.3979 / GSO	-0.3979	82,096
f_{18}	2	-3	-3 / GSO	-3	100,996
f_{19}	3	3.86	3.8628 / GSO	3.8628	160,338
f_{20}	6	3.32	3.2697 / GSO	3.3219	457,836
f_{21}	4	10	7.5439 / PSO	10.1532	251,648
f_{22}	4	10	8.3553 / PSO	10.4029	316,096
f_{23}	4	10	8.9439 / PSO	10.5364	304,312

4 Conclusion

CFO is not nearly as highly developed as GSO, GA and PSO, but it performs very well with nothing more than the objective function as a user input. CFO performed better than or essentially as well as GSO, GA and PSO on 21 of 23 test functions. By contrast, GSO compared to PSO and GA returned the best performance on only 15 benchmarks. CFO's performance thus is better than GSO's, which in turn performed better than PSO and GA on this benchmark suite.

CFO's further development lies in two areas: architecture and theory. While this note describes one CFO implementation that works well, there no doubt is room for considerable improvement. CFO's performance is a sensitive function of the IPD, and there are limitless IPD possibilities. One approach applies precise geometric and combinatorial rules of construction to create function evaluation "centers" in a hypercube using extreme points, faces, and line segments [22-24]. Successively shrinking Ω as described for f_{12} is another possible improvement. Theoretically, gravitationally trapped NEOs may lead to a deeper understanding of the metaheuristic, as might alternative formulations involving the concepts of kinetic or total energy for masses moving under gravity. At this time there is no proof of convergence or complexity analysis for CFO, which is not unusual for new metaheuristics. As discussed in [1,14], for example, the first Ant System proof of convergence appeared four years after its introduction for a "rather peculiar ACO algorithm." Just as ACO was professed on an empirical basis, so too is CFO. Perhaps the CFO-NEO nexus [7] provides a basis for that much desired proof of convergence?

These observations and the results reported here show that CFO merits further study with many areas of potentially fruitful research. Hopefully this note encourages talented researchers to become involved in CFO's further development. For interested readers, a complete source code listing of the program used for this note is available in electronic format upon request to the author (rf2@ieee.org).

References

1. Formato, R.A.: Central Force Optimization: A New Metaheuristic with Applications in Applied Electromagnetics. *Prog. Electromagnetics Research* 77, 425–449 (2007), <http://ceta.mit.edu/PIER/pier.php?volume=77>
2. Formato, R.A.: Central Force Optimization: A New Computational Framework For Multi-dimensional Search and Optimization. In: Krasnogor, N., Nicosia, G., Pavone, M., Pelta, D. (eds.) *Nature Inspired Cooperative Strategies for Optimization (NICSO 2007)*. *Studies in Computational Intelligence*, vol. 129, pp. 221–238. Springer, Heidelberg (2008)
3. Formato, R.A.: Central Force Optimisation: A New Gradient-Like Metaheuristic for Multidimensional Search and Optimisation. *Int. J. Bio-Inspired Computation* 1, 217–238 (2009)
4. Formato, R.A.: Central Force Optimization: A New Deterministic Gradient-Like Optimization Metaheuristic. *OPSEARCH* 46, 25–51 (2009)
5. Qubati, G.M., Formato, R.A., Dib, N.I.: Antenna Benchmark Performance and Array Synthesis using Central Force Optimisation. *IET (U.K.) Microwaves, Antennas & Propagation* 5, 583–592 (2010)

6. Formato, R.A.: Improved CFO Algorithm for Antenna Optimization. *Prog. Electromagnetics Research B*, 405–425 (2010)
7. Formato, R.A.: Are Near Earth Objects the Key to Optimization Theory? arXiv:0912.1394 (2009), <http://arXiv.org>
8. Formato, R.A.: Central Force Optimization and NEOs – First Cousins?. *Journal of Multiple-Valued Logic and Soft Computing* (2010) (in press)
9. Formato, R.A.: NEOs – A Physicomimetic Framework for Central Force Optimization?. *Applied Mathematics and Computation* (review)
10. Formato, R.A.: Central Force Optimization with Variable Initial Probes and Adaptive Decision Space. *Applied Mathematics and Computation* (review)
11. Formato, R.A.: Pseudorandomness in Central Force Optimization, arXiv:1001.0317 (2010), <http://arXiv.org>
12. Formato, R.A.: Comparative Results: Group Search Optimizer and Central Force Optimization, arXiv:1002.2798 (2010), <http://arXiv.org>
13. Formato, R.A.: Central Force Optimization Applied to the PBM Suite of Antenna Benchmarks, arXiv:1003-0221 (2010), <http://arXiv.org>
14. Dorigo, M., Birattari, M., Stützle, T.: Ant Colony Optimization. *IEEE Computational Intelligence Magazine*, 28–39 (November 2006)
15. Campana, E.F., Fasano, G., Pinto, A.: Particle Swarm Optimization: dynamic system analysis for Parameter Selection in global Optimization frameworks, http://www.dis.uniroma1.it/~fasano/Cam_Fas_Pin_23_2005.pdf
16. Hsiao, Y., Chuang, C., Jiang, J., Chien, C.: A Novel Optimization Algorithm: Space Gravitational Optimization. In: *Proc. of 2005 IEEE International Conference on Systems, Man, and Cybernetics*, vol. 3, pp. 2323–2328 (2005)
17. Chuang, C., Jiang, J.: Integrated Radiation Optimization: Inspired by the Gravitational Radiation in the Curvature Of Space-Time. In: *2007 IEEE Congress on Evolutionary Computation (CEC 2007)*, pp. 3157–3164 (2007)
18. Rashedi, E., Nezamabadi-pour, H., Saryazdi, S., Farsangi, M.: Allocation of Static Var Compensator Using Gravitational Search Algorithm. In: *Proc. First Joint Congress on Fuzzy and Intelligent Systems*, Ferdowsi University of Mashad, Iran, pp. 29–31 (2007)
19. Rashedi, E., Nezamabadi-pour, H., Saryazdi, S.: GSA: A Gravitational Search Algorithm. *Information Sciences* 179, 2232–2248 (2009)
20. He, S., Wu, Q.H., Saunders, J.R.: Group Search Optimizer: An Optimization Algorithm Inspired by Animal Searching behavior. *IEEE Tran. Evol. Comp.* 13, 973–990 (2009)
21. CIS Publication Spotlight. *IEEE Computational Intelligence Magazine* 5, 5 (February 2010)
22. Glover, F.: Generating Diverse Solutions For Global Function Optimization (2010), <http://spot.colorado.edu/~glover/>
23. Glover, F.: A Template for Scatter Search and Path Relinking, <http://spot.colorado.edu/~glover/>
24. Omran, M.G.H.: private communication, Dept. of Computer Science, Gulf University for Science & Technology, Hawally 32093, Kuwait