Wireless Engineering and Technology, 2025, 16(1), 1-39
https://www.scirp.org/journal/wet
ISSN Online: 2152-2308
ISSN Print: 2152-2294

Six-Element Yagi-Uda Array: An Example of Using Negative Gravity in Central Force Optimization for Improving Decision Space Exploration

Richard A. Formato

Consulting Engineer & Registered Patent Attorney, Harwich, USA
Email: rf2@ieee.org

How to cite this paper: Formato, R.A. (2025) Six-Element Yagi-Uda Array: An Example of Using Negative Gravity in Central Force Optimization for Improving Decision Space Exploration. *Wireless Engineering and Technology*, **16**, 1-39.
https://doi.org/10.4236/wet.2025.161001

Received: January 22, 2025
Accepted: January 28, 2025
Published: January 31, 2025

Abstract

State-of-the-art antenna design and optimization (D/O) is increasingly being done using Global Search and Optimization (GSO) algorithms such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Differential Evolution (DE), and a plethora of other evolutionary algorithms, among them Central Force Optimization (CFO), which is the subject of this note. CFO analogizes real gravity in the real Universe so that its gravity is usually attractive in nature, that is, "positive". But in metaphorical CFO space the algorithm designer is free to turn gravity on its head by making it negative, and doing so to a small extent can improve CFO's exploration of the search space thus providing even better results. This extension is discussed in some detail and applied to a 6-element Yagi-Uda array as an example.

Keywords

Central Force Optimization, CFO, Optimization, Antenna Design and Optimization, Evolutionary Algorithm, Swarm Algorithm, Metaheuristic, Yagi, Yagi-Uda, Negative Gravity

1. Introduction

Central Force Optimization (CFO) [1]-[3] is a global search and optimization (GSO) metaheuristic based on gravitational kinematics, the motions of real masses whose trajectories are controlled by real gravity. CFO searches a decision space (DS) for the *maxima* of an objective function whose value is its "fitness" and whose topology on DS is unknown or unknowable

The CFO metaphor flies "probes" that sample DS and converge on local extrema. It is inherently *deterministic*, so that every run with the same setup yields precisely the same results. CFO's exploitation of discovered extrema (quickly converging on a local maximum) is quite good [4], but its exploration (sampling unexplored regions of DS) can be inhibited by that very attribute because probes that have coalesced are no longer available to search elsewhere. This dichotomy plagues most GSO's. To quote "'*Exploration* and *exploitation* are the two cornerstones of problem solving by search.' For more than a decade, Eiben and Schippers' advocacy for balancing between these two *antagonistic cornerstones* still greatly influences the research directions of evolutionary algorithms (EAs)…" ([5], emphasis added) and additionally discussed in [6]-[8]. Like all GSO algorithms CFO is subject to the inescapable tension between exploration (adequately sampling DS) and exploitation (quickly converging on global maxima). Using *negative gravity* may mitigate these opposing effects, thereby improving CFO's exploration. Injecting a small amount hopefully discovers new maxima that otherwise would be missed. They may include global maxima or simply other extrema with similar fitness values. This note looks into this possibility with some examples.

CFO has been used both for *design* and for *optimization* (D/O) to solve "real-world" problems, often, but not always, in an engineering setting. *Design* refers to meeting a specified minimum performance, whereas *Optimization* refers to determining the greatest objective function value(s) and their location(s) in DS (the "best" fitnesses and coordinates). Importantly, running any GSO algorithm against "benchmark" functions is altogether different from D/O on typical real world problems. Benchmark functions are known analytically *a priori*, as are their maxima value(s) and their location(s) in DS. Solving benchmarks is quite different from the usual "real world" case of starting without any objective function at all. Therefore, a key element of solving the real world problem is *formulating* a suitable objective function. But accomplishing this can be daunting, especially in complex cases, and using a deterministic algorithm like CFO can make a big difference because there is no inherent randomness in CFO. This note looks into this issue as well.

Most GSO's are stochastic in nature, so that even with the same setup successive runs produce different results. In order to evaluate the effect of a change in the objective function, even a slight change, say, using a different coefficient in some term, requires multiple runs of a stochastic GSO, often tens or hundreds to build sufficient statistics to reliably measure the effect of that new coefficient. In stark contrast, CFO requires only two runs, one with the original objective function and a second with the modified version because *all* changes in the runs' outputs are a consequence *only* of modifying the objective function, not random changes because the GSO itself is inherently stochastic.

CFO has been effectively used with excellent results to solve problems in a wide range of disparate disciplines, among them, as examples: training neural networks [9], patch antenna synthesis [10], power system state space pruning [11], antenna design [12]-[19], arid region water distribution [20], UAV flight path planning [21], bandpass filter design [22], humanoid robot gait [23], ensembles of neural networks [24], satellite image fusion [25], medical image fusion [26], iris recognition

[27], and filter design [28].

This paper is organized as follows: Introduction (§1), Effect of Negative Gravity (§2), Benefit of Negative Gravity (§3), Conclusion (§4), References, Appendix A1 (π Fractions), Appendix A2 (CFO). Note that appendix citations are included in the Reference section.

2. Effect of Negative Gravity

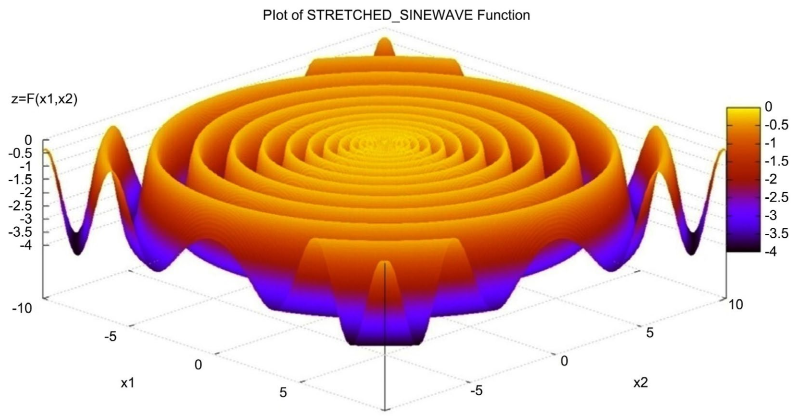
With respect to the sign of CFO's gravity, + or −, the question is, does making it negative benefit CFO's performance, and if so, why, or does it impede it? Positive gravity causes CFO's probes to always move toward greater fitnesses, never away, and consequently to some degree positive gravity inevitably impedes CFO's exploration. In fact, CFO often converges very quickly [4], which is a favorable attribute, but not if rapid exploitation is accomplished at the expense of under-sampling DS, which may well be the case. Adding some negative gravity that causes probes to fly away from each other may address this issue by causing probes that otherwise would coalesce to explore more widely by flying into regions that have been under-sampled or perhaps not sampled at all. The 6-element Yagi test case reported here shows that a small amount of negative gravity indeed does benefit CFO's performance, ostensibly because it enhances CFO's exploration while retaining the algorithm's ability to exploit already located maxima. In the next section a couple of two-dimensional (2D) functions will be used to illustrate the effect of negative gravity which was injected using π-fractions (Appendix A1). π-fractions #0-1,000,001 are downloadable at https://app.box.com/s/qdd8rzrhgaozne0ag1nes9jkm0bj6ark. Those data are provided for any interested user and may be distributed without limitation.

2.1. Stretched Sine Wave

The first function illustrating negative gravity is the stretched Sine Wave [29] defined as

$$f(\vec{x}) = \sum_{i=1}^{N_d-1} \left(x_{i+1}^2 + x_i^2\right)^{0.25} \cdot \left\{\sin^2\left[50\left(x_{i+1}^2 + x_i^2\right)^{0.1}\right] + 0.1\right\}$$

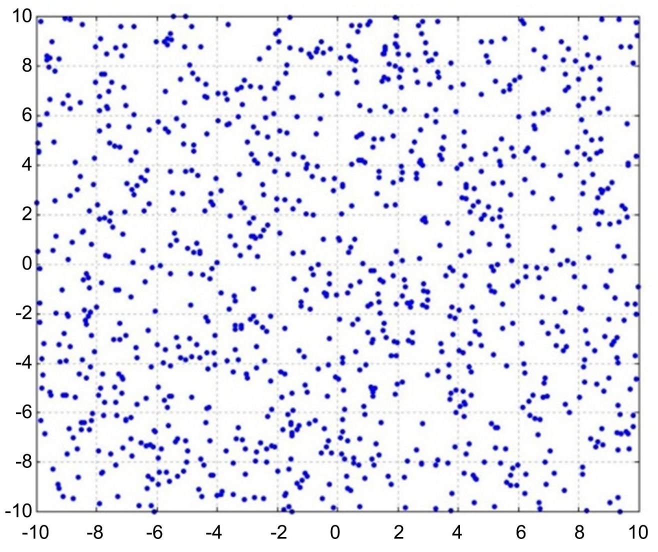
The stretched Sine Wave is plotted below.

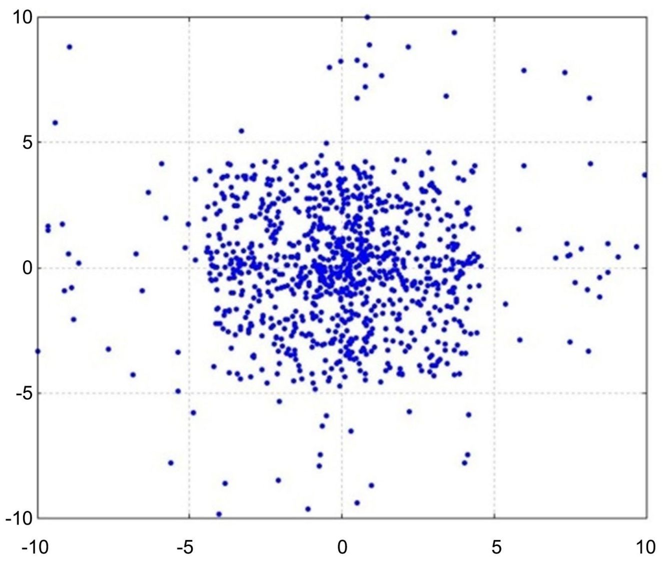


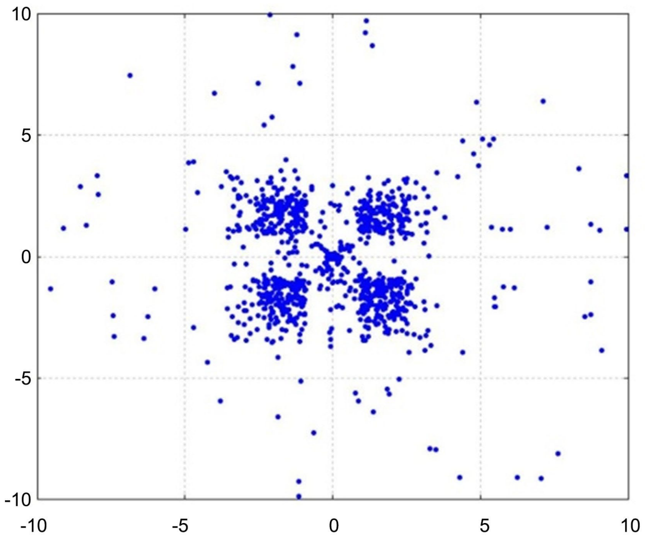
Stretched Sine Wave

For both test functions a dense π-fraction initial probe distribution (IPD) was used to provide a large number of sampling points in order to visualize how CFO's probe distributions evolved with positive and negative gravity by plotting the probe positions as the run progressed. The gravitational constants (Appendix A2) were $G = \pm 2$.

Figure 1 shows the Sine Wave probe evolution with positive gravity at steps 0 (IPD), 2, 4 and 20, in order, top to bottom. This benchmark's maximum is zero at the origin of the 10x10 decision space, and the probes' convergence on the maximum is visually evident. CFO's positive gravity causes the probes to come together around the maximum. Even though the Sine Wave is circularly symmetric DS is not, and the effect of this asymmetry is evident in the probe distributions that cluster along the DS diagonals. Because of the dense IPD, the CFO run was intentionally short (20 steps), and it returned a best fitness of −0.005543 at the point (−0.00288, 0.00106).







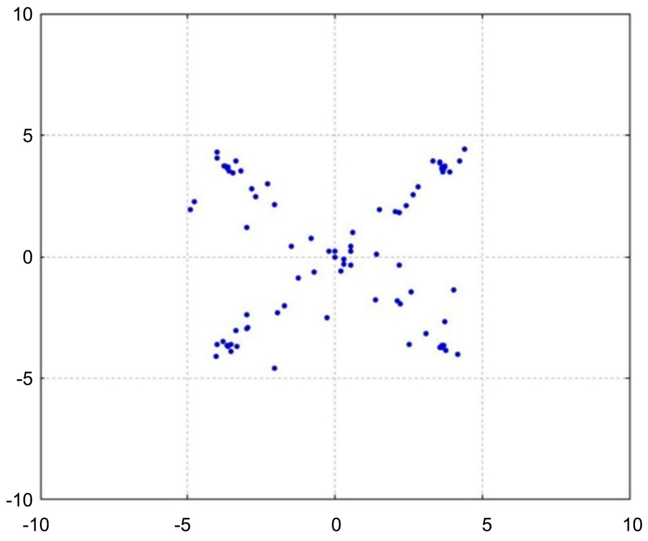
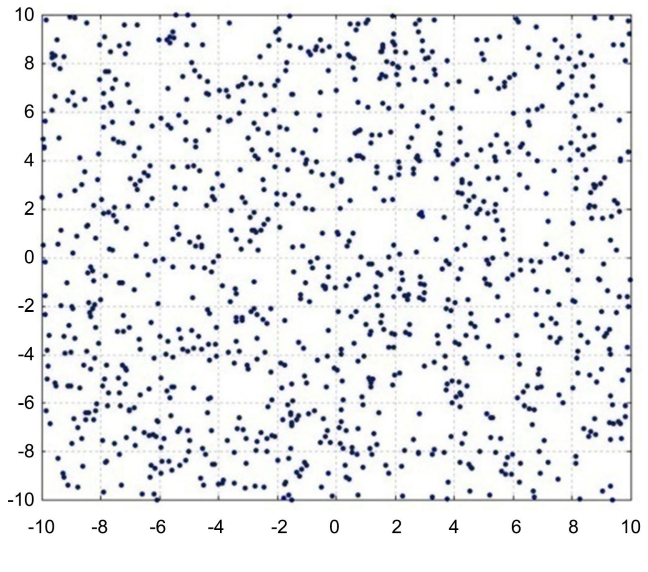


Figure 1. Probes Locations, Stretched Sine Wave, Positive Gravity, $G = +2$.



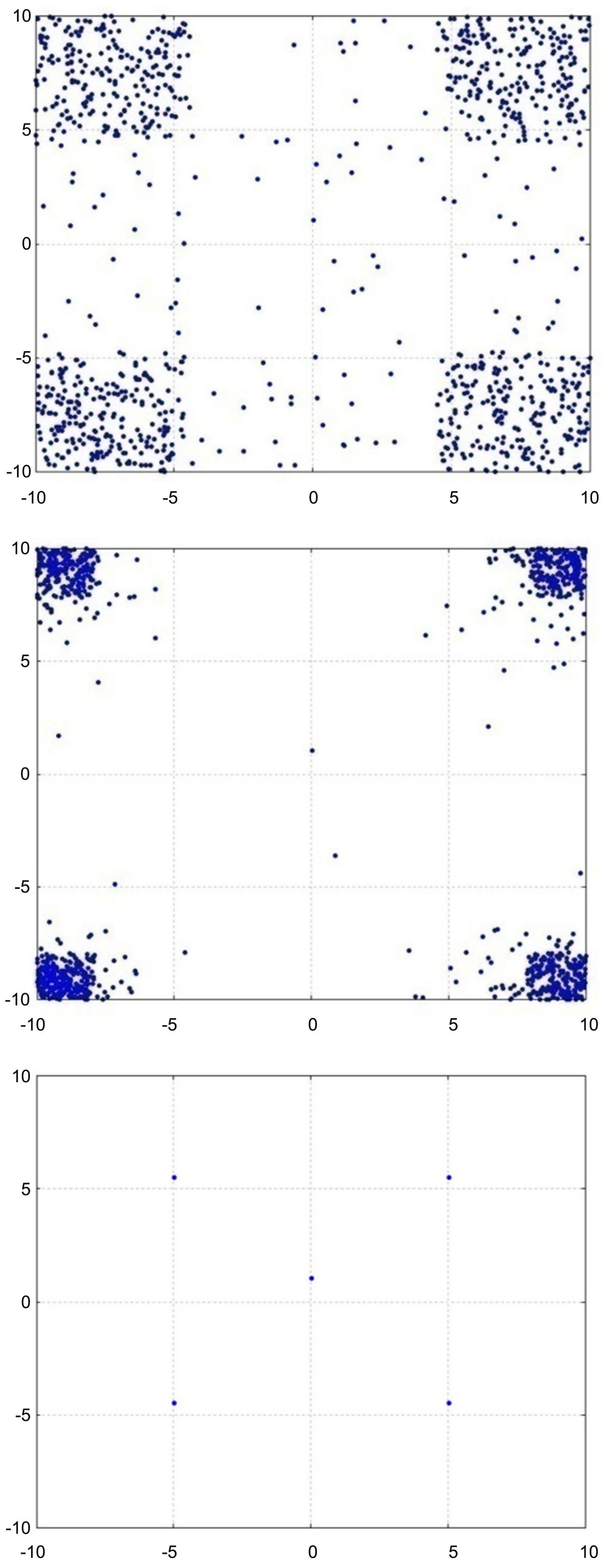


Figure 2. Probes, Stretched Sine Wave, Negative Gravity, $G = -2$.

Figure 2 shows the Sine Wave probe evolution with negative gravity in the same format as Figure 1. The dramatic effect of $G < 0$ is quite apparent—CFO's probes are all forced away from the maximum and instead cluster along the DS diagonals in the far corners, again a result of the Sine Wave/DS asymmetry. The negative gravity flew the probes as far apart as possible. At the end of the run, a best fitness of −0.102165 was returned at the point (0.04672, +1.02846).

2.2. Gaussian Grid

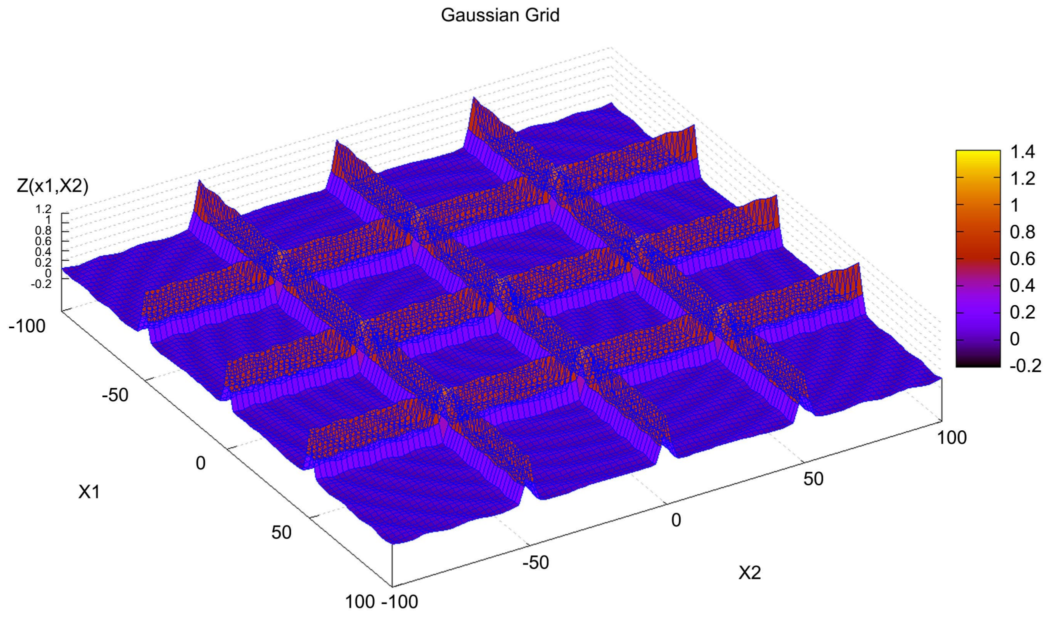
The second negative gravity test function is the Gussian Grid. It provides another compelling example of the validity of CFO's gravitational metaphor and the effect of negative gravity. Note that the Sine Wave is a recognized benchmark whereas the grid is not. An attractive force of gravity requires positive mass and a positive gravitational constant (Appendix A2). Because the CFO implementation employed in this note forces the mass to be positive, the effect of negative gravity is readily demonstrated by setting the gravitational constant G to a negative value, in this case $G = -2$, and as with the Sine Wave the result is quite dramatic.

The two-dimensional Gussian Grid function is defined as

$$f(x_1,x_2) = \min\left\{1, \sum_{i=1}^{3}\left(e^{-\frac{[x_1+50(i-2)]^2}{\sigma^2}} + e^{-\frac{[x_2+50(i-2)]^2}{\sigma^2}}\right)\right\} + \frac{1}{40}\sin\left(\frac{2\pi\sqrt{x_1^2+x_2^2}}{10}\right)$$

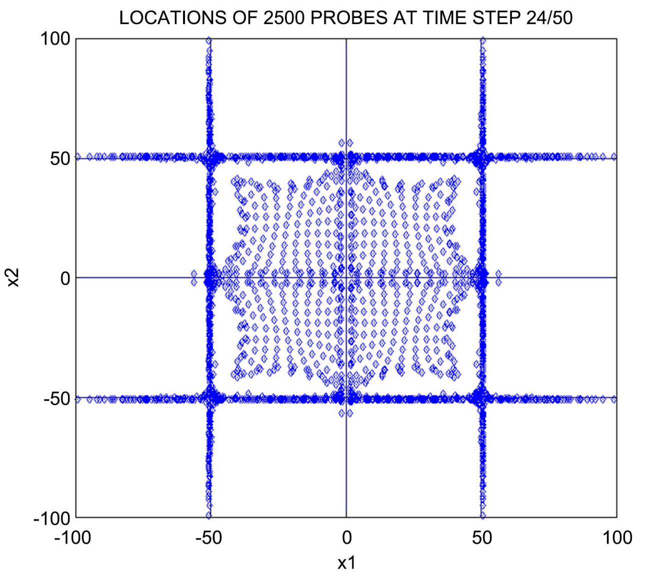
$$-100 \le x_1, x_2 \le 100, \ \ \sigma = 2.$$

While the grid is not a recognized benchmark, it is useful to demonstrate CFO's behavior in distributing probes with and without negative gravity. The grid is plotted below. Its global maxima of 1.025 lie on the grid lines $x_1 = 0, \pm 50$; $x_2 = 0, \pm 50$.

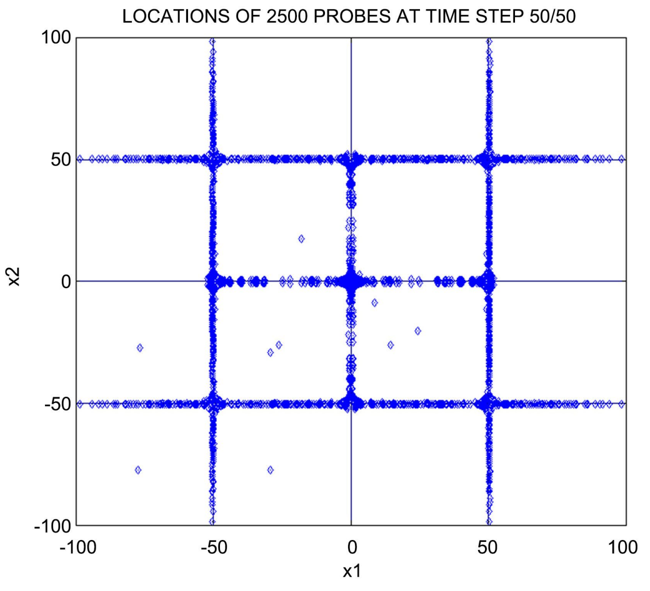


Gaussian Grid

In order to visualize the probes' evolution as the run progresses, CFO again was run with a dense IPD, in this case a uniform grid of probes. **Figure 3** shows the probe distribution at steps 24 and 50 with positive gravity, $G = +2$. At step 24 the grid's structure is quite evident, and by step 50 it is fully resolved. Interestingly, the grid lines passing through the origin do not contain probes for $\left|x_{1,2}\right| > 50$. The likely reason is that probes on these segments were attracted to the large probe concentration near the origin. This thinning effect also seems to be occurring at step 50 on the segments $-50 \le x_1 \le 50$, $x_2 = 0$ and $-50 \le x_2 \le 50$, $x_1 = 0$ where the probe density near the segment center is noticeably lower than at the ends.



(a)



(b)

Figure 3. (a) Step 24 with $G = +2$; (b) Step 50 with $G = +2$.

With G < 0 the probe distribution at step 50 (end of run) is shown in **Figure 4**, in stark contrast to the positive gravity case. Instead of clustering along the grid lines, the probes are symmetrically forced away from maxima to the very edges of the decision space. With negative gravity CFO's probes fly away from each other instead of clustering near the grid's maxima. As an interesting corollary, the grid's landscape contains a continuum of local maxima, but in decision spaces like this, that is ones containing an uncountable or a very large number of maxima, most GSO's converge on only one. In many real-world problems, however, that single maximum may not actually be the "best". In real-world D/O, especially engineering design, identical or nearly identical fitnesses do not necessarily correspond to fungible designs. As a practical matter, of many solutions with the same or nearly the same fitness, usually one is better than the others for reasons that may not be quantifiable in an objective function. For example, one design may be less expensive, or easier to fabricate, or to maintain, or to distribute, and so on with regard to any number of ancillary considerations that are not or cannot be easily reflected in an objective function. In cases where there are many indistinguishable or nearly indistinguishable maxima, a deterministic GSO like CFO, with positive gravity and a small amount of negative gravity may be useful in exploring DS as a preprocessor to aid in locating many maxima, not just one, thus expanding the range of solutions.

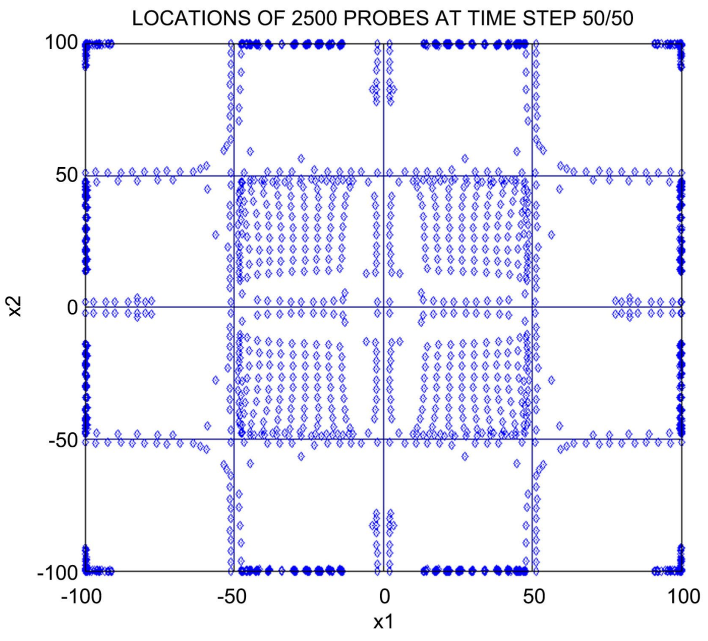


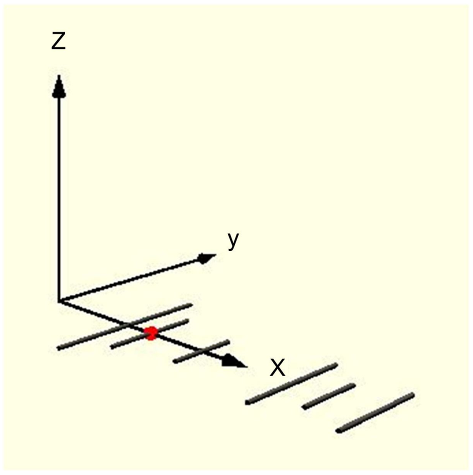
Figure 4. Step 50 with $G = -2$.

3. Example of Negative Gravity Benefit, a "Real-World" Problem: 6-Element Yagi-Uda Array

The beneficial effects of injecting a small amount of negative gravity in CFO were discussed in considerable detail in some previously published papers [30] [31].

Highlights from that work are presented here to supplement the discussion so far. While the example used here is drawn from Yagi antenna design, the basic idea of adding some negative gravity to CFO applies to any CFO problem. The reader is not expected or required to have any background in antennas or electromagnetics and should feel free to quickly peruse this section if desired.

The structure under consideration is the six-element Yagi-Uda array shown below. Arrays of this type are used across the entire radio spectrum and range in size from sub-centimeter to tens of meters. The antenna comprises six parallel dipole elements mounted on a common axis ("boom"). One element is connected to the radio transmitter/receiver ("driven", marked by the colored dot) while the others are not ("parasitic"). The antenna's performance over a range of frequencies ("bandwidth", BW) is measured primarily by how much energy it radiates in specific directions ("gain") and how well it accepts power from a radio transmitter ("standing wave ratio", SWR).

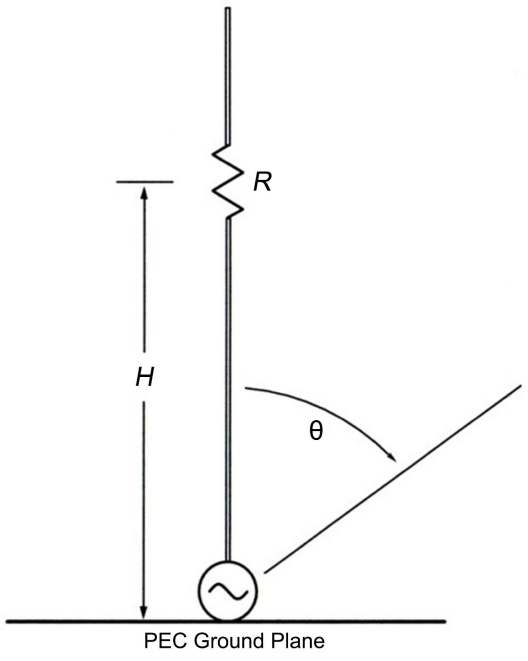


The problem here is to maximize an objective (fitness) function that combines the following parameters: antenna gain, SWR and BW. The specific mathematical form of the fitness function is not known *a priori*. It must be determined by the designer. There is a limitless number of ways these parameters can be mathematically combined, and not all of them are suitable as measures of how well a particular Yagi design performs. The Yagi problem is eighteen-dimensional (diameter/length of each element, 12 variables); five spacings along the boom; and "input impedance" that determines (SWR). The very first question is, what mathematical combination of these eighteen variables does the best job of measuring how good a particular design is? Not any easy question!

3.1. Formulating a Fitness Function

As pointed out previously, formulating a suitable fitness function can be quite difficult, but using a deterministic GSO like CFO makes the job much easier. This point is important enough that it will be further examined with a very simple example: a base-fed monopole. Its simplicity notwithstanding, even in a case like this, specifying a suitable fitness function can be problematic.

The objective here is maximizing the bandwidth of this simplest possible antenna structure: a base-fed 10.2-meter monopole (a straight wire) on an infinite perfectly conducting ground plane as shown below. The monopole's bandwidth will be increased by adding a fixed resistor *R* at a height *H*. A GSO must determine the best values of *R* and *H*. Unlike the Yagi problem, this problem is only two-dimensional (2D) so that its fitness function can be visualized. Of course, as is usually the case, the problem statement doesn't come with the required objective function. The designer must formulate it from scratch. What is a suitable form? There are many parameters that can be included, among them the monopole complex input impedance, R_{in}, X_{in}, its radiation efficiency, ε, (reduced by adding *R*), the system characteristic impedance, Z_0, and the maximum gain, $G_{\max}$, all across a range of desired frequencies (3 - 30 Mhz in this case). The algorithm designer is free to define *any* objective function that works well to measure the monopole's fitness by combining these parameters in *any* desired manner.



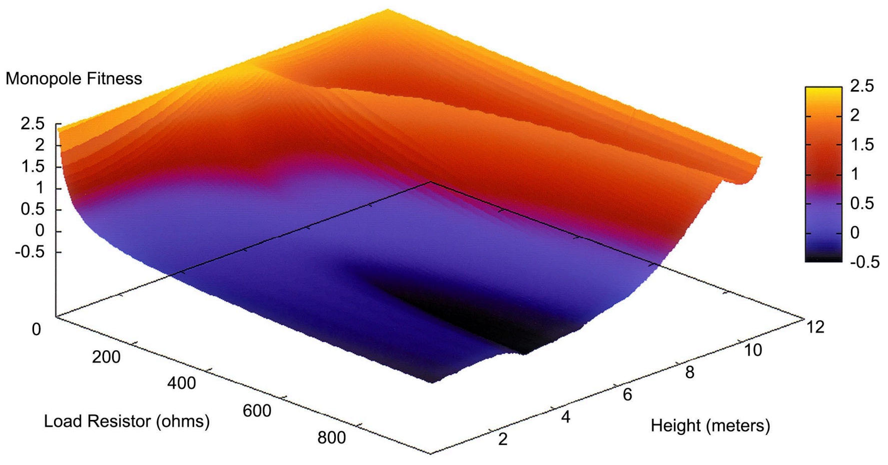
To illustrate the difficulty in formulating a suitable objective function, three forms are considered for this problem, each one ostensibly suitable for the problem at hand. The candidates are:

$$f_1(R,H)=\frac{\min(\varepsilon)+\min(G_{\max})}{\mathrm{SWR}}$$

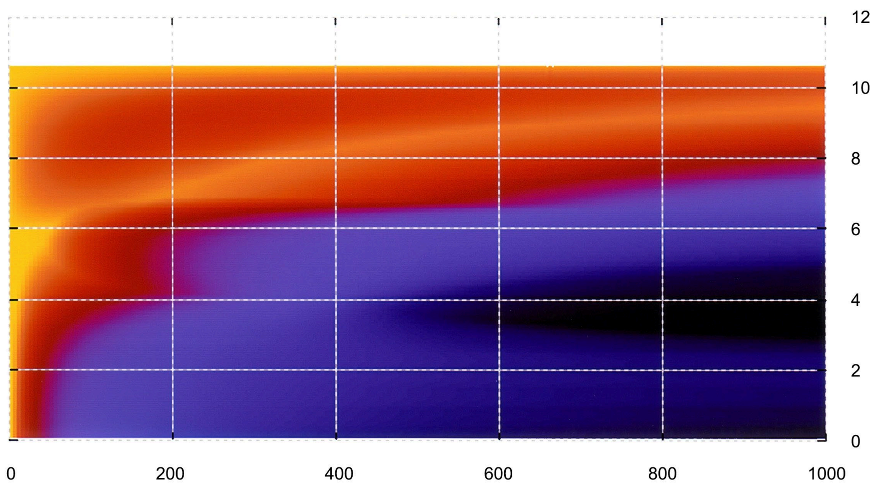
$$f_2(R,H,Z_0)=\frac{\min(\varepsilon)}{|Z_0-\max(R_{in})|\cdot|\max(X_{in})|}$$

$$f_3(R,H,Z_0)=\frac{\min(\varepsilon)}{|Z_0-\max(R_{in})|\cdot\Delta\mathrm{SWR}\cdot[\max(X_{in})-\min(X_{in})]}$$

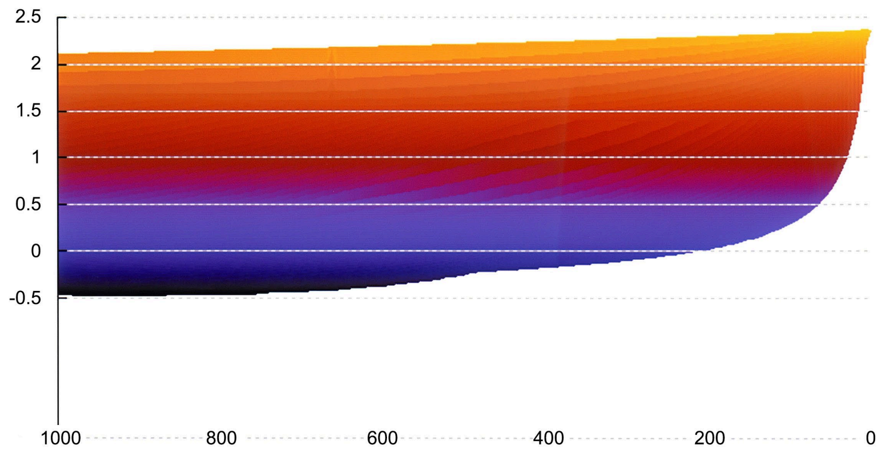
Their 2D landscapes are plotted below in **Figures 5-7** (principal plane views). Visual inspection reveals that two of the three are poor choices, one of them extremely poor. The landscape for function #1, shown in **Figure 5**, is smoothly varying. The landscape for function #3, which is shown in **Figure 7**, also is smoothly varying. Both of these candidate functions are unimodal, which generally improves a GSO's effectiveness. But in stark contrast, function f_2, whose topology is shown in **Figure 6**, is an especially bad choice because it is pathologically spikey, in the extreme, even though at first blush its functional form seems to be good for maximizing BW. The fact is, f_2 simply is so pathological that its spikey nature would make it difficult, if not impossible, for most GSO's to locate its global maximum. Of course, without being able to visualize these functions, as is done here, there is no way of knowing how poor a choice f_2 actually is. Returning to the two possibilities that do make sense, f_1 has a narrow maximum not too different from surrounding fitnesses, and it is quite close to the DS boundary, which may make the maximum difficult to locate. By comparison, f_3 has a well-defined global maximum that is well within DS. Of the three candidates, there is no question that f_3 clearly is the best.



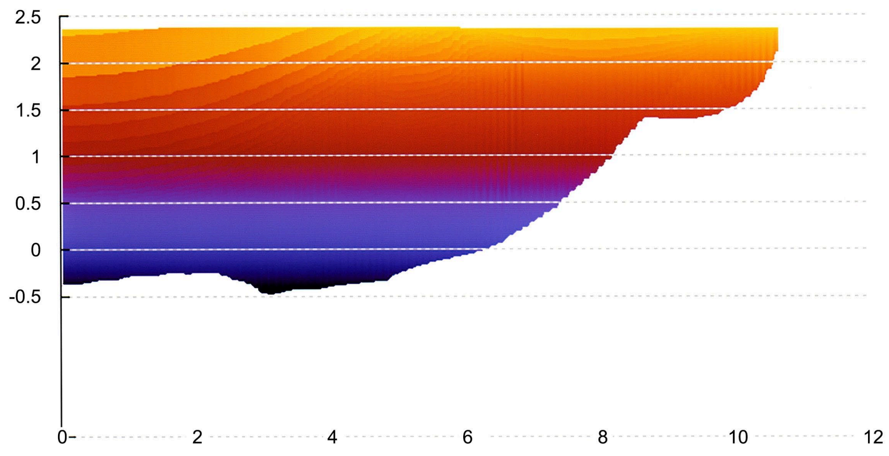
(a)



(b)

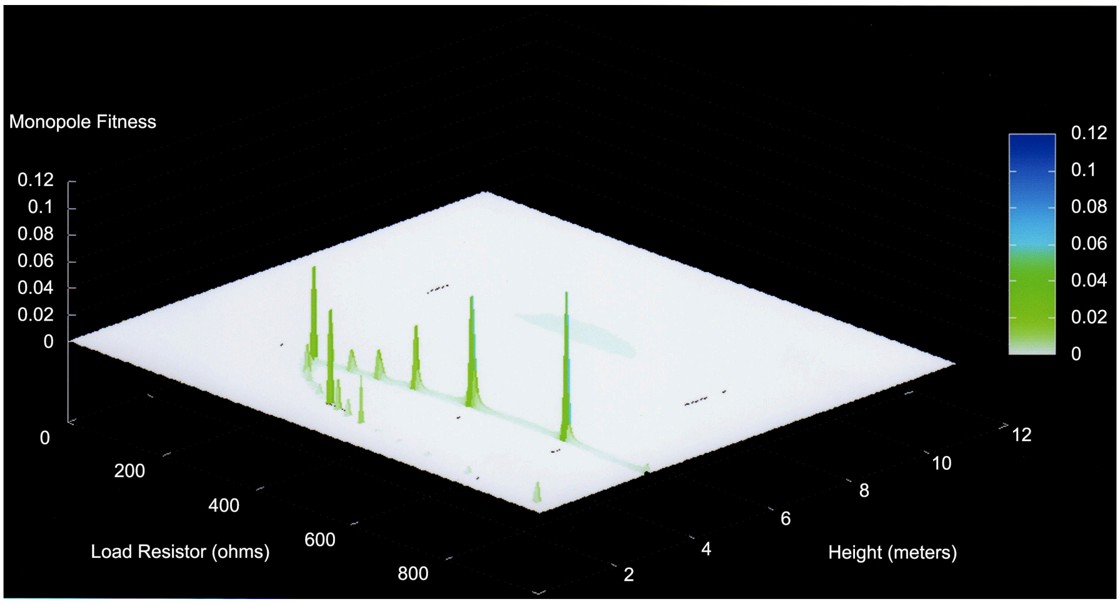


(c)

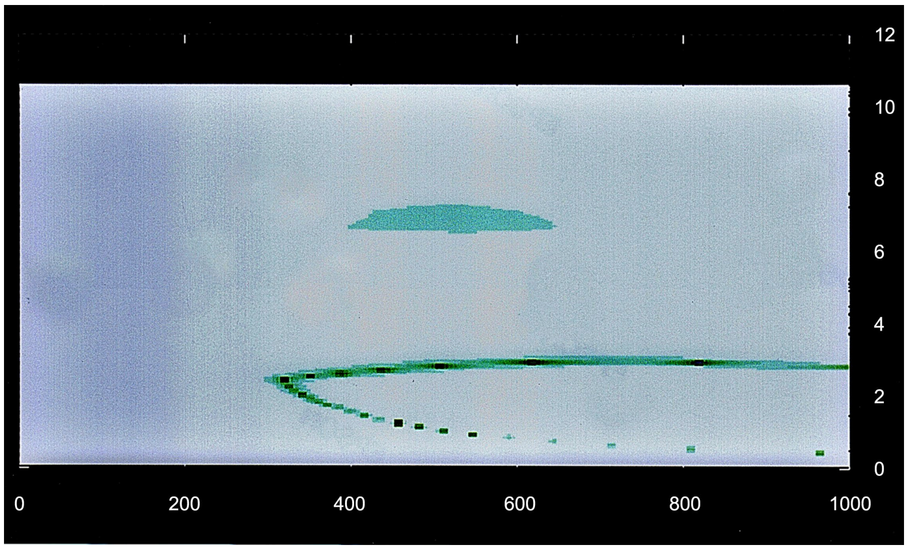


(d)

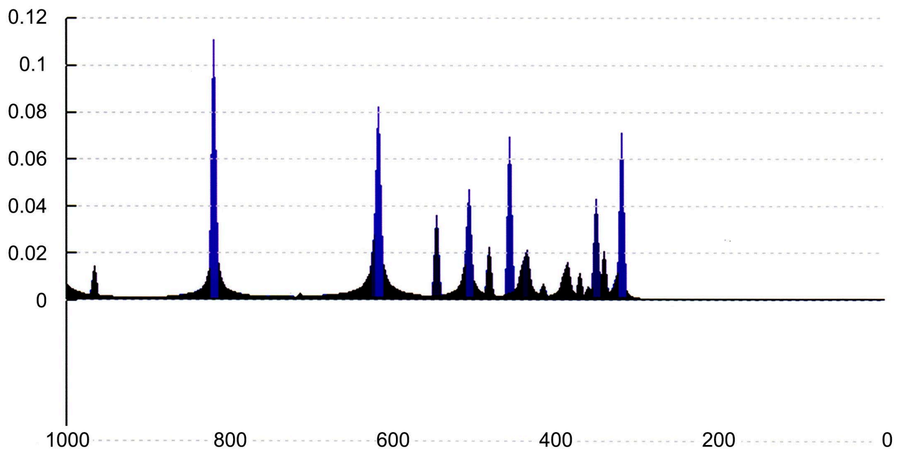
Figure 5. Landscape for objective function $f_1(R,H,50)$.



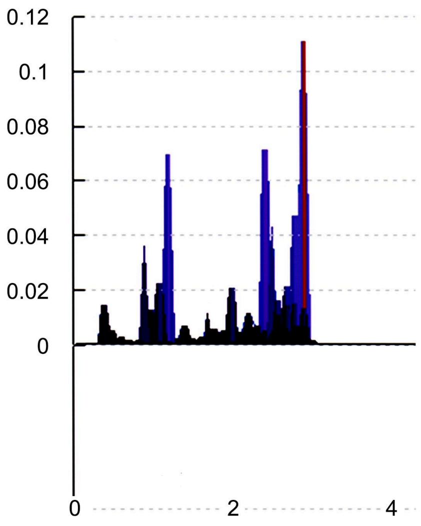
(a)



(b)

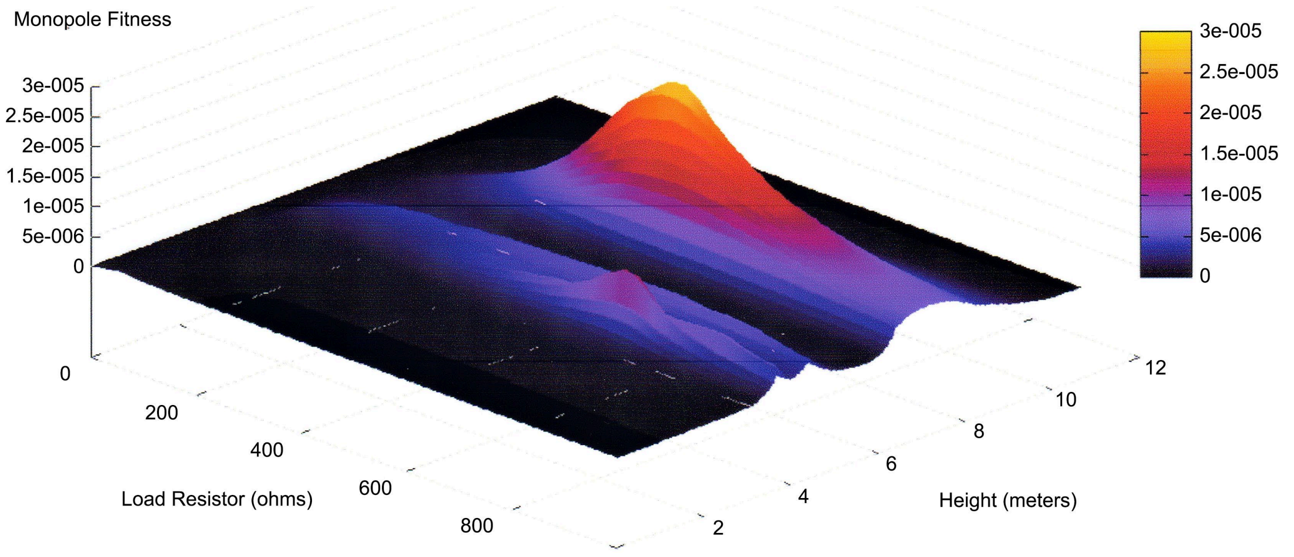


(c)

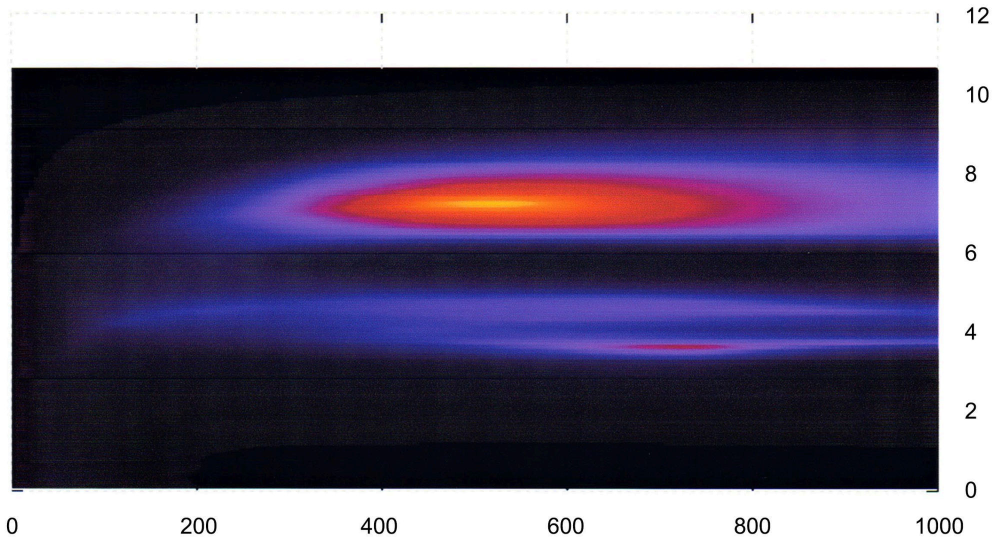


(d)

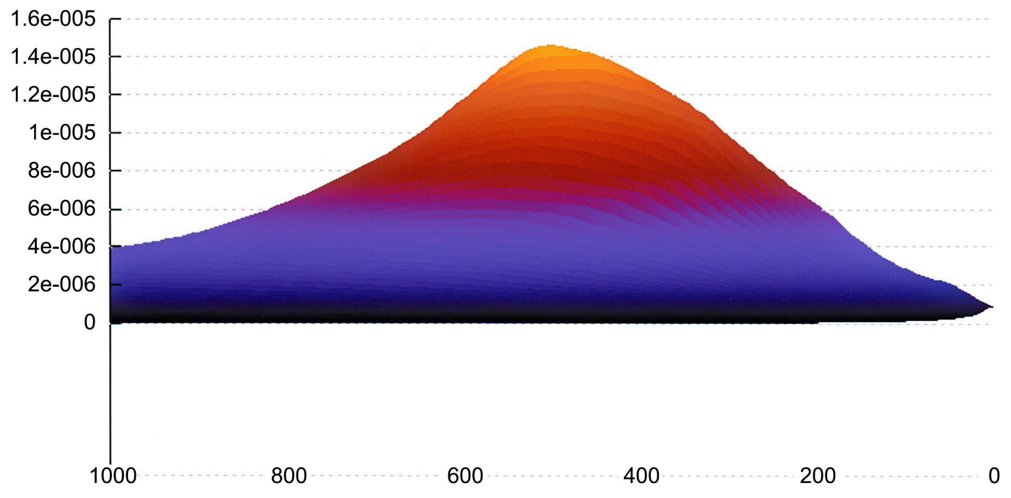
Figure 6. Landscape for Objective Function $f_2(R,H,50)$.


Monopole Fitness
3e-005
2.5e-005
2e-005
1.5e-005
1e-005
5e-006
0
0
200
400
600
800
Load Resistor (ohms)
2
4
6
8
10
12
Height (meters)
3e-005
2.5e-005
2e-005
1.5e-005
1e-005
5e-006
0

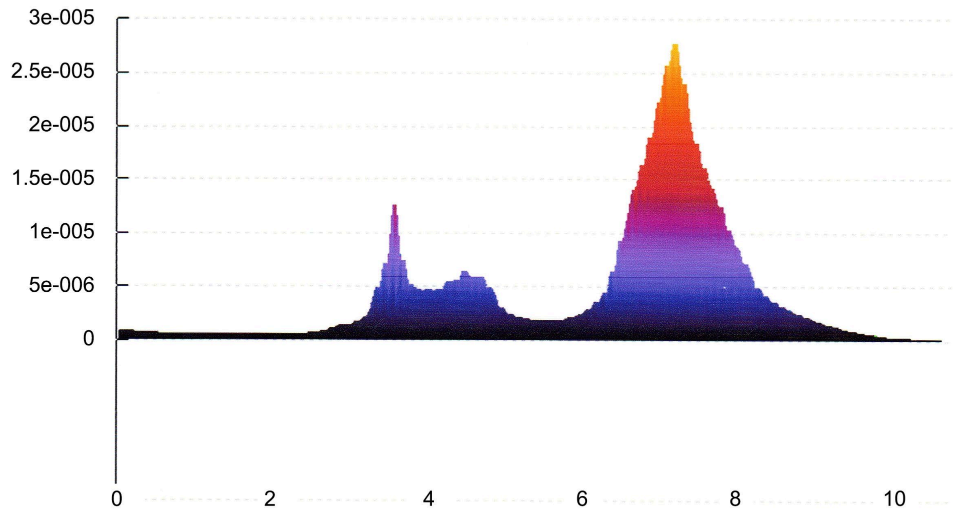
(a)


12
10
8
6
4
2
0
0
200
400
600
800
1000

(b)


1.6e-005
1.4e-005
1.2e-005
1e-005
8e-006
6e-006
4e-006
2e-006
0
1000
800
600
400
200
0

(c)



(d)

Figure 7. Landscape for Objective Function $f_3(R,H,50)$.

So what happens if candidate objective functions cannot be visualized? How can the algorithm designer formulate a suitable fitness if, as an example, instead of just R and H another component is added, say, an inductor, whose value and location are to be determined the same way R and H must be found? Now the problem is 4D instead of 2D. The landscape for candidate objective functions cannot be plotted. How can the suitability of candidate functions be assessed? Using a deterministic GSO like CFO makes all the difference because changes in the algorithm's results from one run to the next are *entirely* a result of changes to the fitness function, and nothing else, for example randomness in the GSO itself. A deterministic GSO permits direct comparison of how well candidate objective functions serve to actually solve the problem at hand, whereas stochastic GSO's simply cannot without making many runs.

3.2. Yagi Array Results

Getting back to the Yagi problem, its objective was to maximize the array's impedance bandwidth and forward gain over a specified frequency range. After trying several fitness functions, the following simple form was settled on:

$$\begin{aligned}\text{YAGI Fitness} &= c_1 * \text{Gain}(\text{L}) + c_3 * \text{Gain}(\text{M}) + c_5 * \text{Gain}(\text{U}) \\ &\quad - c_2 * \text{SWR}(\text{L}) - c_4 * \text{SWR}(\text{M}) - c_6 * \text{SWR}(\text{U})\end{aligned}$$

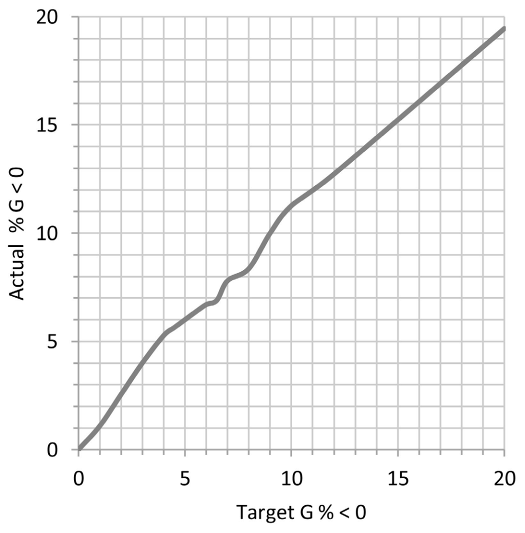
where L/M/U are the lower/mid/upper frequencies at which the Yagi's power gain and SWR are computed. The weighting coefficients are: $c_1 = c_2 = c_5 = c_6 = 1$; $c_3 = c_4 = 3$, which intentionally favors midband performance, slightly. Their values were determined empirically by making successive CFO runs using different coefficient values and evaluating the results of each one, something that could not be done quickly with any stochastic GSO.

Negative gravity was injected into CFO as follows. At each step negative gravity was pseudo randomly assigned using π-fractions (data files available, see Appendix A1). At each step in the CFO run, the value of the π-fraction was tested against the target level of negative gravity and the sign of gravity adjusted accordingly (positive or negative). CFO pseudocode and the actual vs. target amount of negative gravity are shown below in Figure 8(a) and Figure 8(b), more detail in [30] [31]. This algorithm is referred to as "CFO/NG".

```
Alg: CFO - Six-Element Yagi Design

Initialization (Step 0)*
        (i)     Initialize $\pi$-fraction index ($\leftarrow$ 2)
        (ii)    Compute Initial Probe Distribution($\pi$-frac prv's)
        (iii)   Compute Initial Fitness Matrix
        (iv)    Initial Probe Accelerations ($\leftarrow$ 0)
For Step#: 1 - #Steps
        (i)     Pseudo randomly assign positive or
                negative gravity ($\pi$-frac prv's)
        (ii)    Compute Probe Position Vectors
        (iii)   Retrieve Errant Probes
        (iv)    Compute Fitness Matrix for
                Current Probe Distribution
        (v)     Compute Accelerations from Current
                Probe Distribution/Fitnesses
        (vi)    Adjust Repositioning Factor $F_{rep}$
Next
```

(a)



(b)

Figure 8. (a) CFO-Negative Gravity Pseudocode; (b) CFO Negative Gravity using π Fractions.

The Yagi fitness and its maximum gain as a function of the amount of negative gravity are shown in Figure 9 and Figure 10. With zero negative gravity, the fitness and max gain, respectively, are about 48 and 11.4 dB, which are both very respectable values. Adding a small amount of negative gravity at first reduces these values, but when the negative gravity level reaches about six percent both the fitness and gain increase dramatically, and they remain above their initial values with as much as about 10% negative gravity, after which they gradually decrease. These two plots alone convincingly demonstrate that adding some small amount of negative gravity can help quite a bit. The negative gravity allows CFO's probes to explore more widely and to discover even better array designs. The improved fitness and gain correspond to solutions that were missed by CFO without negative gravity because CFO failed to fly probes into their regions of DS. In fact, combining $G < 0$ in CFO with a couple of other extensions (Dynamic Threshold Optimization and Elitism) has resulted in more than 19% improvement in the fitness of a 6-element Yagi [30] [31]. There is no question that some small amount of negative gravity can be beneficial.

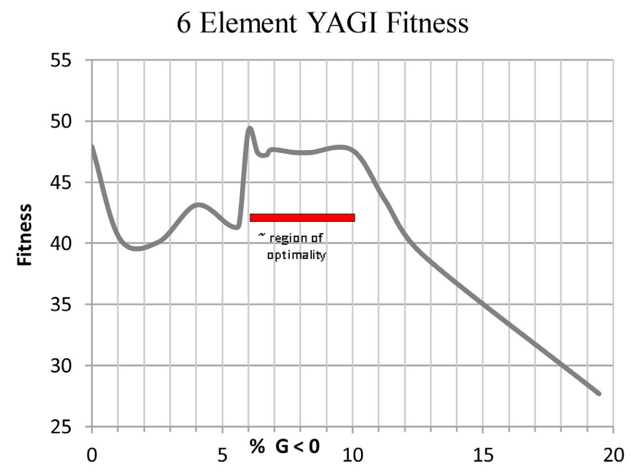


Figure 9. Yagi Fitness vs. Degree of Negative Gravity.

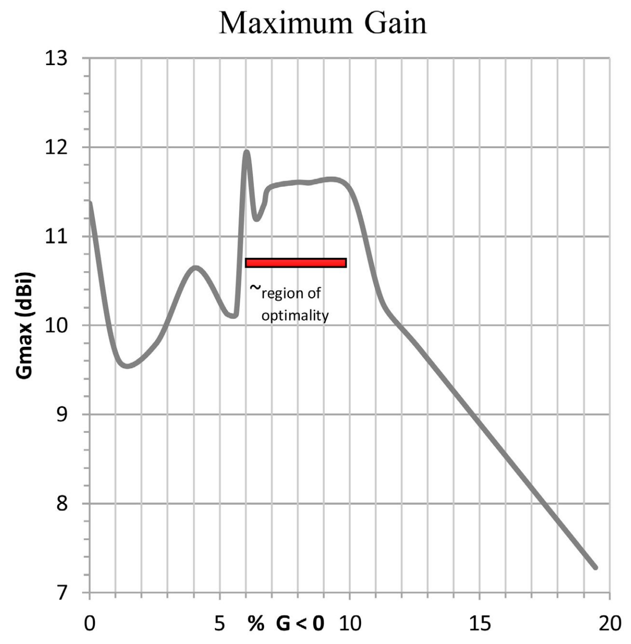


Figure 10. Yagi Maximum Gain vs. Degree of Negative Gravity.

For comparison, Figure 11 shows the Yagi gain optimized with several other GSO's, two CFO variants and the original GSO that resulted in using a 6-element Yagi array as the reference design, namely, *Dominating Line Cone Search*, "DLCS" (curve labeled "A3" in Figure 11) [32]. Additional details are available in [33]. The DLCS maximum gain is barely over 10 dBi, whereas the CFO/NG value at 6% negative gravity is nearly 12 dBi, a very substantial improvement of nearly 20%! For this particular problem, 6% negative gravity provides the best maximum gain as is evident from the plot, which presents data for negative gravity levels between zero and just under 20%. If CFO/NG is being used as the GSO, it is apparent that runs should be made with varying amounts of negative gravity because the results are sensitive to how much is injected. Of course, because CFO is deterministic, as is pi-fraction-injected negative gravity, the observed changes in CFO's output can only be a result of adding negative gravity if all other CFO parameters remain unchanged, which is the case here. And because the only change is adding negative gravity, all other CFO parameters remaining fixed, for example, the magnitude of the gravitational constant (only its sign was modified), there is no issue with sensitivity to run parameters, which can be a major concern with stochastic GSO's such as Particle Swarm or Ant Colony Optimization.

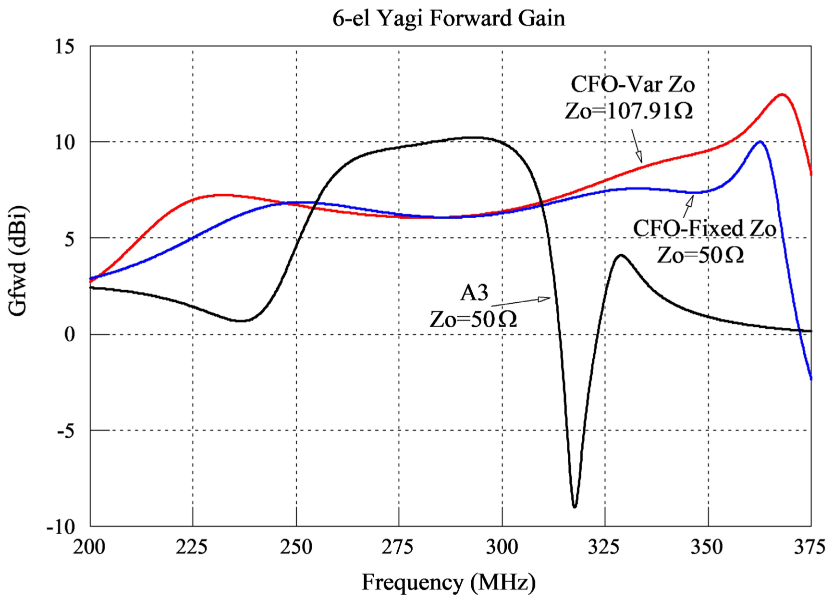


Figure 11. Yagi Maximum Gain with Other GSO's.

4. Conclusion

It is apparent that injecting a small amount of negative gravity into Central Force Optimization can produce substantially better results because the algorithm's exploration of DS is improved. The examples discussed here are compelling. It is reasonable to speculate that injecting a small amount of negative gravity will enhance CFO cross the board, that is, *any* CFO run against *any* GSO problem. Of course, how well the negative gravity CFO extension works is likely to be problemdependent, that is, providing better results on some than on others. It is therefore recommended that the negative gravity approach be used in all CFO implementations because there is no significant downside to running CFO/NG instead of CFO alone.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Formato, R.A. (2007) Central Force Optimization: A New Metaheuristic with Applications in Applied Electromagnetics. *Progress In Electromagnetics Research*, **77**, 425-491. https://doi.org/10.2528/pier07082403

[2] Formato, R.A. (2008) Central Force Optimization: A New Nature Inspired Computational Framework for Multidimensional Search and Optimization. In: Krasnogor, N., Nicosia, G., Pavone, M. and Pelta, D., Eds., *Studies in Computational Intelligence*, Springer, 221-238. https://doi.org/10.1007/978-3-540-78987-1_21

[3] Formato, R.A. (2009) Central Force Optimisation: A New Gradient-Like Metaheuristic for Multidimensional Search and Optimisation. *International Journal of Bio-Inspired Computation*, **1**, 217-238. https://doi.org/10.1504/ijbic.2009.024721

[4] Ding, D.S., Luo, X.P., Chen, J.F., Wang, X.J., Du, P.Y. and Guo, Y.F. (2011) A Convergence Proof and Parameter Analysis of Central Force. *Journal of Convergence Information Technology*, **6**, 16-23.

[5] Črepinšek, M., Liu, S. and Mernik, M. (2013) Exploration and Exploitation in Evolutionary Algorithms. *ACM Computing Surveys*, **45**, 1-33. https://doi.org/10.1145/2480741.2480752

[6] Luke, S. (2015) Essentials of Metaheuristics. 2nd Edition. https://cs.gmu.edu/~sean/book/metaheuristics/

[7] Korošec, P. and Eftimov, T. (2020) Insights into Exploration and Exploitation Power of Optimization Algorithm Using DSC Tool. *Mathematics*, **8**, Article 1474. https://doi.org/10.3390/math8091474

[8] Cuevas, E., Echavarría, A. and Ramírez-Ortegón, M.A. (2013) An Optimization Algorithm Inspired by the States of Matter That Improves the Balance between Exploration and Exploitation. *Applied Intelligence*, **40**, 256-272. https://doi.org/10.1007/s10489-013-0458-0

[9] Green, R.C., Wang, L. and Alam, M. (2012) Training Neural Networks Using Central Force Optimization and Particle Swarm Optimization: Insights and Comparisons. *Expert Systems with Applications*, **39**, 555-563. https://doi.org/10.1016/j.eswa.2011.07.046

[10] Qubati, G.M. and Dib, N.I. (2010) Microstrip Patch Antenna Optimization Using Modified Central Force Optimization. *Progress In Electromagnetics Research B*, **21**, 281-298. https://doi.org/10.2528/pierb10050511

[11] Green, R.C., Wang, L.F. and Alam, M. (2012) Intelligent State Space Pruning with Local Search for Power System Reliability Evaluation. 2012 3*rd IEEE PES Innovative Smart Grid Technologies Europe* (*ISGT Europe*), Berlin, 14-17 October 2012, 1-8. https://doi.org/10.1109/isgteurope.2012.6465775

[12] Mahmoud, K.R. (2011) Central Force Optimization: Nelder-Mead Hybrid Algorithm for Rectangular Microstrip Antenna Design. *Electromagnetics*, **31**, 578-592.

https://doi.org/10.1080/02726343.2011.621110

[13] Asi, M. and Dib, N.I. (2010) Design of Multilayer Microwave Broadband Absorbers Using Central Force Optimization. *Progress In Electromagnetics Research B*, **26**, 101-113. https://doi.org/10.2528/pierb10090103

[14] Montaser, A.M., Mahmoud, K.R. and Elmikati, H.A. (2012) Integration of an Opti-Mized E-Shaped Patch Antenna into a Laptop Structure for Bluetooth and Notched-UWB Standards Using Optimization Techniques. *Applied Computational Electromagnetics Society*, **27**, 784.

[15] Montaser, A.M., Mahmoud, K.R., Abdel-Rahman, A.B. and Elmikati, H.A. (2013) Design Bluetooth and Notched-UWB E-Shape Antenna Using Optimization Techniques. *Progress In Electromagnetics Research B*, **47**, 279-295. https://doi.org/10.2528/pierb12101407

[16] Montaser, A.M., Mahmoud, K.R. and Elmikati, H.A. (2012) B15. Tri-Band Slotted Bow-Tie Antenna Design for RFID Reader Using Hybrid CFO-NM Algorithm. 2012 29*th National Radio Science Conference* (*NRSC*), Cairo, 10-12 April 2012, 119-126. https://doi.org/10.1109/nrsc.2012.6208515

[17] Gonzalez, J.E., Amaya, I. and Correa, R., (2013) Design of an Optimal Multi-Layered Electromagnetic Absorber through Central Force Optimization. *Progress in Electromagnetics Research Symposium*, Stockholm, 12-15 August 2013, 1082.

[18] Ahmed, E., R. Mahmoud, K., Hamad, S. and T. Fayed, Z. (2013) CFO Parallel Implementation on GPU for Adaptive Beam-Forming Applications. *International Journal of Computer Applications*, **70**, 10-16. https://doi.org/10.5120/12013-8001

[19] Montaser, A.M., Mahmoud, K.R., Abdel-Rahman, A.B. and Elmikati, H.A. (2013) B10. Design of a Compact Tri-Band Antenna for RFID Handheld Applications Using Optimization Techniques. 2013 30*th National Radio Science Conference* (*NRSC*), Cairo, 16-18 April 2013, 82-89. https://doi.org/10.1109/nrsc.2013.6587905

[20] Haghighi, A. and Ramos, H.M. (2012) Detection of Leakage Freshwater and Friction Factor Calibration in Drinking Networks Using Central Force Optimization. *Water Resources Management*, **26**, 2347-2363. https://doi.org/10.1007/s11269-012-0020-6

[21] Chen, Y.B., Yu, J.Q., Mei, Y.S., Wang, Y.F. and Su, X.L. (2016) Modified Central Force Optimization (MCFO) Algorithm for 3D UAV Path Planning. *Neurocomputing*, **171**, 878-888. https://doi.org/10.1016/j.neucom.2015.07.044

[22] Abdel-Rahman, A.B., Montaser, A.M. and Elmikati, H.A. (2012) Design a Novel Bandpass Filter with Microstrip Resonator Loaded Capacitors Using CFO-HC Algorithm. *The* 2*nd Middle East Conference on Antennas and Propagation*, Cairo, 29-31 December 2012, 1-6. https://doi.org/10.1109/mecap.2012.6618188

[23] Huan, T.T. and Anh, H.P.H. (2019) Optimal Stable Gait for Nonlinear Uncertain Humanoid Robot Using Central Force Optimization Algorithm. *Engineering Computations*, **36**, 599-621. https://doi.org/10.1108/ec-03-2018-0154

[24] Chao, M., Xin, S.Z. and Min, L.S. (2014) Neural Network Ensembles Based on Copula Methods and Distributed Multiobjective Central Force Optimization Algorithm. *Engineering Applications of Artificial Intelligence*, **32**, 203-212. https://doi.org/10.1016/j.engappai.2014.02.009

[25] Talal, T.M., Attiya, G., Metwalli, M.R., Abd El-Samie, F.E. and Dessouky, M.I. (2020) Satellite Image Fusion Based on Modified Central Force Optimization. *Multimedia Tools and Applications*, **79**, 21129-21154. https://doi.org/10.1007/s11042-019-08471-7

[26] El-Hoseny, H.M., Abd El-Rahman, W., El-Rabaie, E.M., Abd El-Samie, F.E. and

Faragallah, O.S. (2018) An Efficient DT-CWT Medical Image Fusion System Based on Modified Central Force Optimization and Histogram Matching. *Infrared Physics & Technology*, **94**, 223-231. https://doi.org/10.1016/j.infrared.2018.09.003

[27] Shaikh, N.F. and Doye, D.D. (2014) A Novel Iris Recognition System Based on Central Force Optimization. *International Journal of Tomography and Simulation*, **27**, 23-34.

[28] A.khaleel, A.A.J. (2009) Linear Phase Finite-Impulse Response Filter Design Using Central Force Optimization Algorithm. Master of Science (M.Sc.) Electrical Engineering (Thesis), University of Jordan Library. https://theses.ju.edu.jo/Original_Abstract/JUF0706056/JUF0706056.pdf

[29] Momin, J. and Yang, X-S. (2033) A literature survey of benchmark functions for Global Optimisation Problems. *International Journal Mathematical Modelling and Numerical Optimization*, **4**, 150-194.

[30] Formato, R.A. (2021) Central Force Optimization with Gravity < 0, Elitism, and Dynamic Threshold Optimization: An Antenna Application, 6-Element Yagi-Uda Arrays. *Wireless Engineering and Technology*, **12**, 53-82. https://doi.org/10.4236/wet.2021.124004

[31] Formato, R.A. (2021) Six-Element Yagi Array Designs Using Central Force Optimization with Pseudo Random Negative Gravity. *Wireless Engineering and Technology*, **12**, 23-51. https://doi.org/10.4236/wet.2021.123003

[32] Lisboa, A.C., Vieira, D.A.G., Vasconcelos, J.A., Saldanha, R.R. and Takahashi, R.H.C. (2009) Monotonically Improving Yagi-Uda Conflicting Specifications Using the Dominating Cone Line Search Method. *IEEE Transactions on Magnetics*, **45**, 1494-1497. https://doi.org/10.1109/tmag.2009.2012688

[33] Formato, R.A. (2012) Improving Bandwidth of Yagi-Uda Arrays. *Wireless Engineering and Technology*, **3**, 18-24. https://doi.org/10.4236/wet.2012.31003

[34] Formato, R.A. (2017) Determinism in Electromagnetic Design & Optimization-Part II: BBP-Derived π Fractions for Generating Uniformly Distributed Sampling Points in Global Search and Optimization Algorithms. *Forum for Electromagnetic Research Methods and Application Technologies*, 19, Article No. 10. http://www.e-fermat.org/

[35] De Rainville, F., Gagné, C., Teytaud, O. and Laurendeau, D. (2012) Evolutionary Optimization of Low-Discrepancy Sequences. *ACM Transactions on Modeling and Computer Simulation*, **22**, 1-25. https://doi.org/10.1145/2133390.2133393

[36] De Rainville, F., Gagné, C., Teytaud, O. and Laurendeau, D. (2009) Optimizing Low-Discrepancy Sequences with an Evolutionary Algorithm. *Proceedings of the* 11*th Annual Conference on Genetic and Evolutionary Computation*, Canada, 8-12 July 2009, 1491-1498. https://doi.org/10.1145/1569901.1570101

[37] Pant, M., Thangaraj, R., Grosan, C. and Abraham, A. (2008) Improved Particle Swarm Optimization with Low-Discrepancy Sequences. 2008 *IEEE Congress on Evolutionary Computation* (*IEEE World Congress on Computational Intelligence*), Hong Kong, 1-6 June 2008, 3011-3018. https://doi.org/10.1109/cec.2008.4631204

[38] Imamichi, T., Numata, H., Mizuta, H. and Ide, T. (2011) Nonlinear Optimization to Generate Non-Overlapping Random Dot Patterns. *Proceedings of the* 2011 *Winter Simulation Conference* (*WSC*), Phoenix, AZ, 11-14 December 2011, 2414-2425.

[39] Alabduljabbar, A.A., Milanovi, J.V. and Al-Eid, E.M. (2008) Low Discrepancy Sequences Based Optimization Algorithm for Tuning PSSs. *Proceedings of the* 10*th International Conference on Probablistic Methods Applied to Power Systems*, Rincon, 25-29 May 2008, 1-9.

[40] Krykova, I. (2003) Evaluating of Path-Dependent Securities with Low Discrepancy

Methods. M.S. Thesis (Financial Mathematics), Worcester Polytechnic Institute.

[41] Jørgensen, K. (2008) Himalaya Options: An Analysis of Pricing Himalaya Options in a Monte Carlo and Quasi-Random Monte Carlo framework. Master's Thesis, Aarhus University.

[42] Bailey, D., Borwein, P. and Plouffe, S. (1997) On the Rapid Computation of Various Polylogarithmic Constants. *Mathematics of Computation*, **66**, 903-913. https://doi.org/10.1090/s0025-5718-97-00856-9

[43] Formato, R.A. (2017) Determinism in Electromagnetic Design & Optimization-Parts I & II. *Forum for Electromagnetic Research Methods and Application Technologies*, **19**, Article No. 9. https://efermat.github.io/

[44] Sörensen, K. (2013) Metaheuristics—The Metaphor Exposed. *International Transactions in Operational Research*, **22**, 3-18. https://doi.org/10.1111/itor.12001

[45] Aranha, C., and Campelo, F. (2023) Lessons from the *Evolutionary Computation Bestiary*. *Artificial Life*, **29**, 421-432. https://github.com/fcampelo/EC-Bestiary

[46] van der Corput, J.G. (1935) Verteilungsfunktionen. *Proceedings Nederlandse Akademie van Wetenschappen*, **38**, 813-821.

[47] Halton, J.H. (1960) On the Efficiency of Certain Quasi-Random Sequences of Points in Evaluating Multi-Dimensional Integrals. *Numerische Mathematik*, **2**, 84-90. https://doi.org/10.1007/bf01386213

[48] Li, W.T., Shi, X.W., Hei, Y.Q., Liu, S.F. and Zhu, J. (2010) A Hybrid Optimization Algorithm and Its Application for Conformal Array Pattern Synthesis. *IEEE Transactions on Antennas and Propagation*, **58**, 3401-3406. https://doi.org/10.1109/tap.2010.2050425

[49] Pehlivanoglu, Y.V. (2013) A New Particle Swarm Optimization Method Enhanced with a Periodic Mutation Strategy and Neural Networks. *IEEE Transactions on Evolutionary Computation*, **17**, 436-452. https://doi.org/10.1109/tevc.2012.2196047

Appendix A1. π FRACTIONS

GSO Sampling: (This material adapted from [34]). Uniformity in randomly generated sample points is an important consideration in GSO. Sample points should be generated using a truly uniformly distributed random variable (rv) calculated from a probability distribution, but most if not all pseudorandom sequence generators fall short because their points in fact are not uniformly distributed. One alternative approach is using Low Discrepancy Sequences (LDS) which are becoming more popular. For example, De Rainville *et al.* [35] [36] provide a summary of the uniformity problem and develop an evolutionary optimization approach for generating LDS. Pant *et al.* [37] describe an improved Particle Swarm Optimization (PSO) algorithm utilizing van der Corput and Sobol LDS. Other representative, not exhaustive, examples include LDS applied to liquid crystal display dot patterns [38], power system stabilizers [39], and, quite interestingly, financial analysis [40] [41].

This appendix describes an alternative approach to generating uniformly distributed sample points using π Fractions computed from hexadecimal digit extraction from the mathematical constant π. Pi Fractions [42] are uniformly distributed and provide a basis for creating reproducible, deterministic sample point distributions that can be used in any GSO algorithm regardless of its fundamental nature, stochastic, deterministic or hybrid. The importance of determinism in electromagnetics problems is discussed in [43].

Another reason for considering π Fractions is what some practitioners, including the author, consider a willy-nilly proliferation of stochastic metaheuristics of questionable merit [44]. Examples range from "Anarchic societies" to "Zombies" [45]. Are these algorithms any good? How can they be efficiently compared head-to-head or to other well-established algorithms? Making them deterministic would be a good first step, and π Fractions can do that. For interested readers, π fractions#0 - 1,000,001 are downloadable at:
https://app.box.com/s/qdd8rzrhgaozne0ag1nes9jkm0bj6ark

BBP Algorithm: The Bailey-Borwein-Plouffe (BBP) algorithm quite remarkably extracts hexadecimal digits from the numerical constant π beginning at *any* digit without having to compute *any* of the preceding digits. BBP is based on the identity

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right)$$

whose derivation and use in BBP are described in detail in [42]. As an example, the hex digits of π starting at digit 1,000,000 are 26C65E52CB459350050E4BB1 and the corresponding π Fraction is 0.151464362347971272412488292131. For all practical purposes the first 215,829 π Fractions are uniformly distributed on [0, 1) with a mean value of 0.499283729688375. The Cumulative Distribution Function (CDF) for these data is plotted in **Figure A1** (1000 bins) in which $\Pr\{\pi_i \le X\}$ is the probability that π Fraction π_i is less than or equal to $0 \le X \le 1$. It is reasonable to speculate that all sequential π_i are uniformly distributed as well. Testing on various subsets of the 215,829 data set reveals a uniform distribution regardless of how many contiguous fractions are included in any fairly large sample or where the sequence is begun. It also seems reasonable to believe that any sufficiently large set of arbitrarily selected π_i also will be uniformly distributed in [0, 1). These characteristics have not been investigated for the other constants discussed in [42], so it is not known whether or not they exhibit similar behavior.

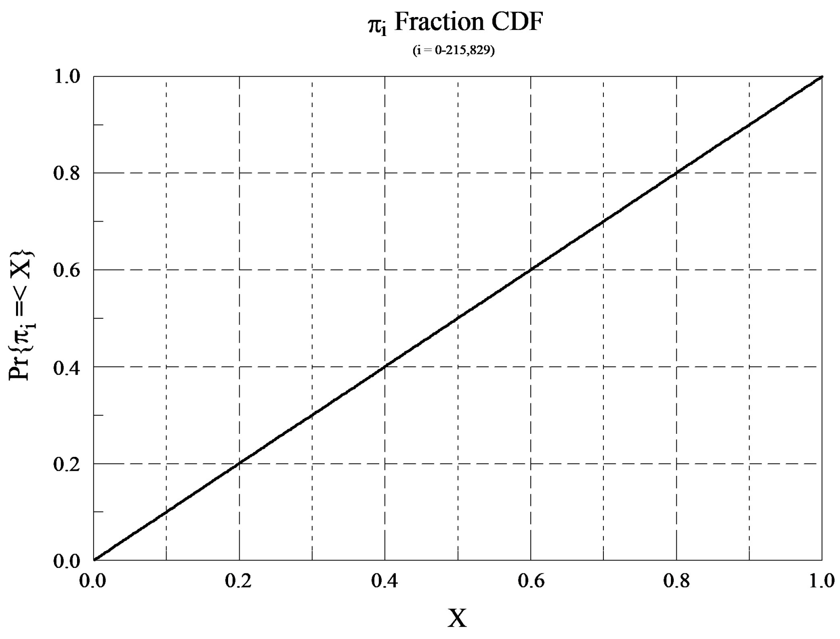


Figure A1. π Fraction CDF.

__Dimensional Correlations:__ Nonuniformity in LDS sequences often is evident in bidimensional plots in high dimensionality spaces. Figure 2 and Figure 3 in [46] are good examples. They show, respectively, almost perfect linear correlations in monotonically increasing van der Corput sequences and correlations between dimensions 7 and 8 in a Halton sequence [46] [47]. Other striking visual examples appear in [40] and [41]. Testing of van der Corput and Halton sequences reveals many undesirable correlations. A typical 30-dimensional Halton example for coordinates 27 and 28 appears Figure A2.

Sample points based on π Fractions also can exhibit strong linear correlations, but apparently only under very limited circumstances. For example, Figure A3 plots (x_{27}, x_{28}) for 1000 points in 30 dimensions using the π Fractions in their order of occurrence (index increment = 1 starting with the first π Fraction). The linear correlation is obvious, but it disappears completely when instead dimensions 27 and 29 are compared as seen in Figure A4. Many test runs suggest that the π Fractions exhibit correlation *only* in successive dimensions and *only* when accessed in their order of occurrence, regardless of where the sequence starts. But when a different index greater than 1 is used, for example, a value of 2, there is no obvious correlation as shown in the (x_{27}, x_{28}) plot in Figure A5. These data suggest it is reasonable to believe that indeed the π Fractions provide uniformly distributed uncorrelated sample points as long as successive fractions are *not* used to compute the sample point coordinates.

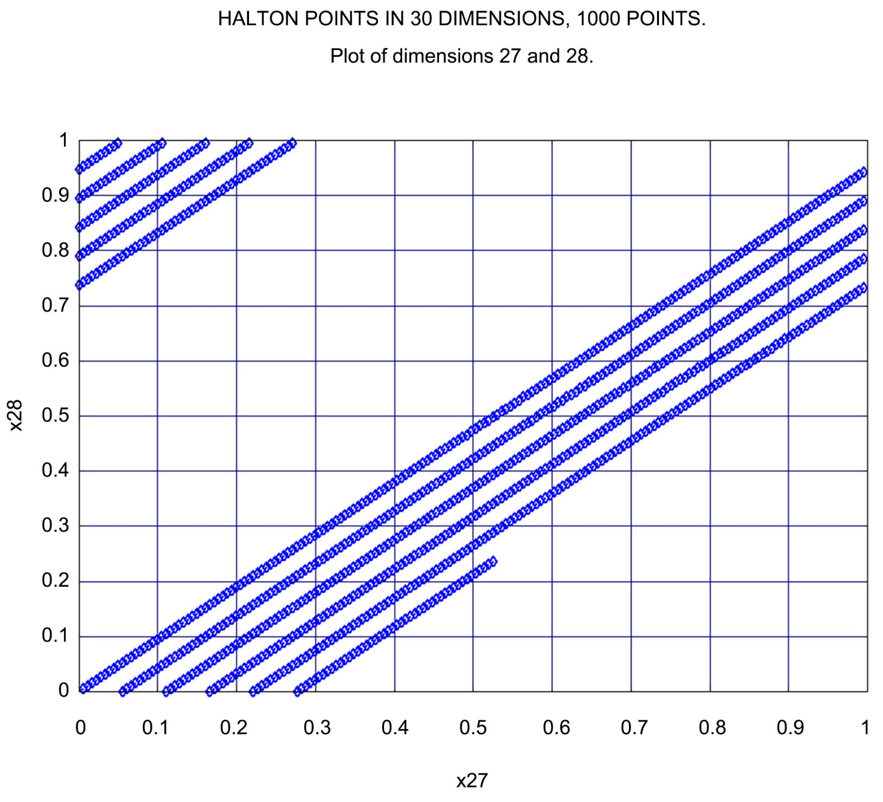


Figure A2. Coordinates (x_{27}, x_{28}) 30D Halton sequence (1000 points).

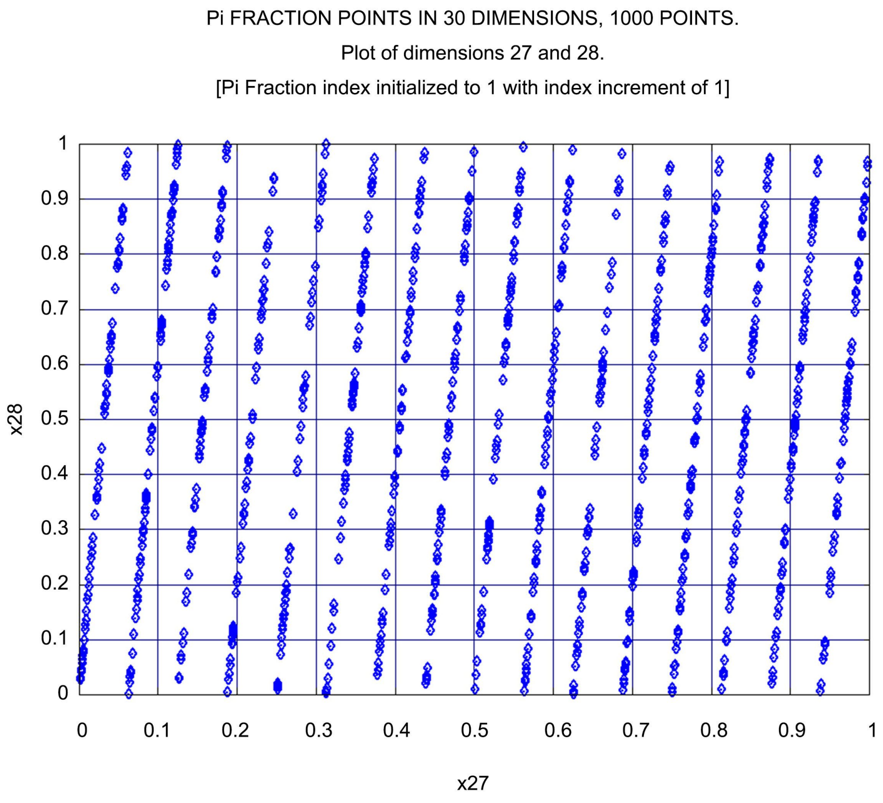


Figure A3. Coordinates (x_{27}, x_{28}) 30D π Fractions, index increment = 1.

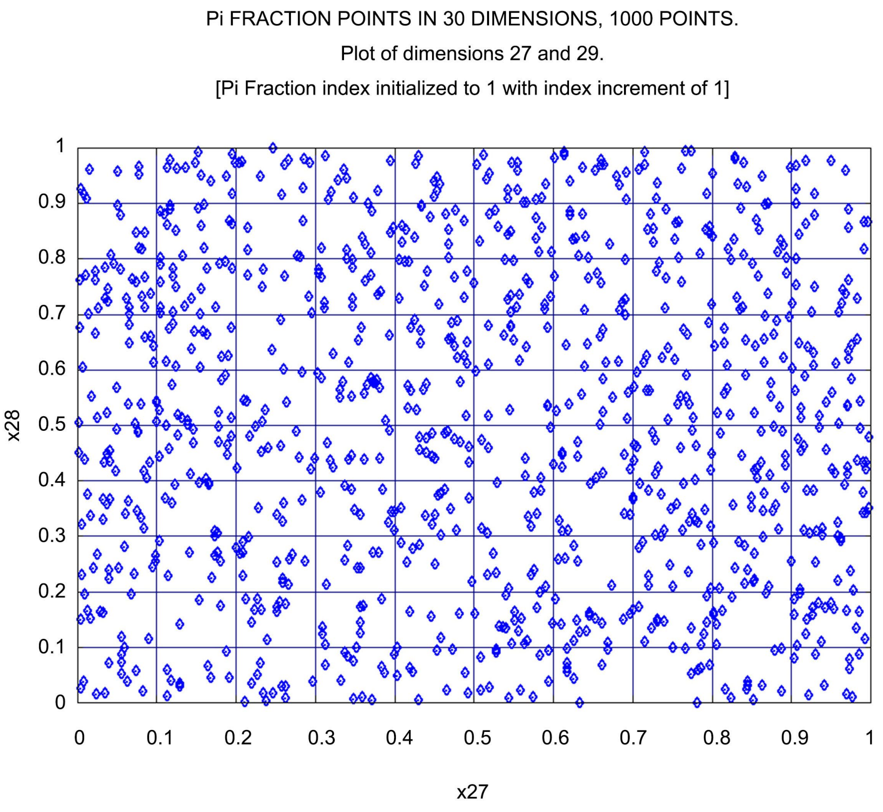


Figure A4. Coordinates (x_{27}, x_{29}) 30D π Fractions, index increment = 1.

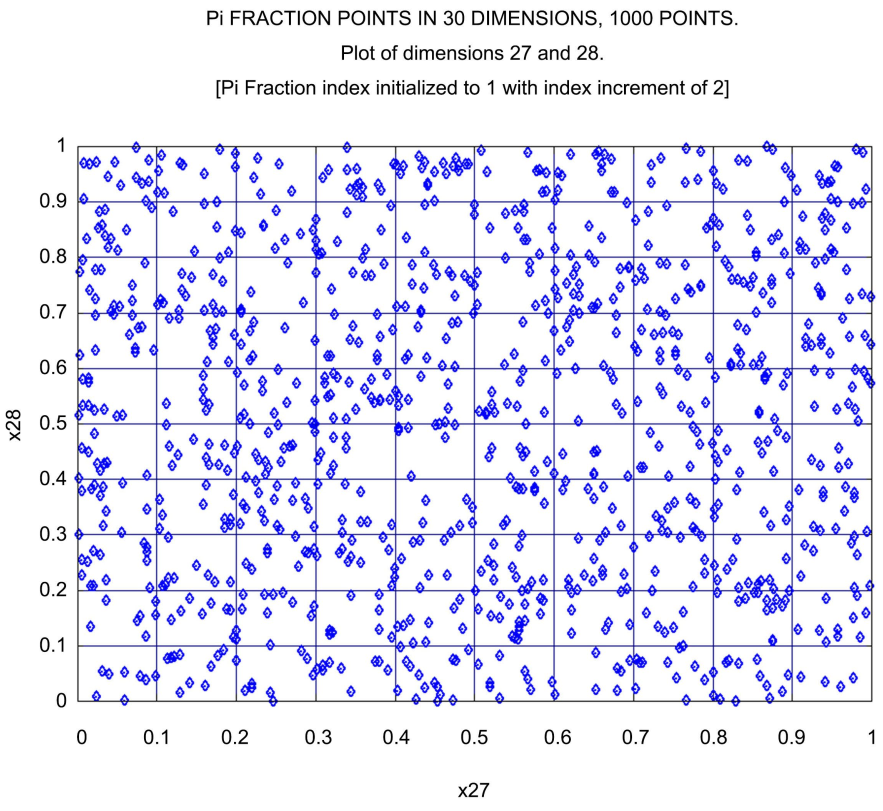


Figure A5. Coordinates (x_{27}, x_{28}) 30D π Fractions, index increment = 2.

An Example - πGASR Algorithm: The utility of π Fractions was investigated by generating uniformly distributed sample points and pseudorandom numbers in the genetic algorithm πGASR which is based on Li *et al.*'s novel GA [48]. A standard GA is improved in by 1) allowing competition between child chromosomes in a new crossover operator resulting in better interpolation and extrapolation of decision space sample points and 2) introducing an iteration-dependent mutation operator. Li *et al.*'s algorithm is referred to here as "Genetic Algorithm with Sibling Rivalry" (GASR) because of the new crossover operator (see [48] for details and note that the "SR" descriptor is introduced here). Its implementation using π Fractions is algorithm πGASR. π Fractions are used to create the initial chromosome distribution and in testing for crossover, mutation, and elitism. Details of the scheme selecting the π_i are determined by the algorithm designer and in this case appear in the source code listing (https://github, search term "PiFractions"). The manner in which the πGASR's fractions are selected avoids the bidimensional correlation issue discussed above. Note that πGASR, like CFO, maximizes the objective function instead of minimizing it.

Benchmark Results

πGASR was tested against the six-function benchmark suite shown in Table A1. Results are reported in Table A2 and Table A3. In Table A1 DS is the decision space, x^* the location of the objective function's known maximum, and $f(x^*)$ its value. This suite was used in [49] to test the new algorithm vibrational-PSO, "v-PSO".

Table A1. v-PSO Benchmark Functions.

Fnc#	*Function*	$f(x)$	DS	x^*	$f(x^*)$
f_1	Ackley	$20\exp\left(-0.2\sqrt{\frac{1}{N_d}\sum_{i=1}^{N_d}x_i^2}\right)+\exp\left(\frac{1}{N_d}\sum_{i=1}^{N_d}\cos(2\pi x_i)\right)-20-e$	$[-30,30]^{N_d}$	$[0]^{N_d}$	0
f_2	Cosine Mixture	$-\sum_{i=1}^{N_d}x_i^2+0.1\sum_{i=1}^{N_d}\cos(5\pi x_i)$	$[-1,1]^{N_d}$	$[0]^{N_d}$	$0.1N_d$
f_3	Exponential	$\exp\left(-0.5\sum_{i=1}^{N_d}x_i^2\right)$	$[-1,1]^{N_d}$	$[0]^{N_d}$	1
f_4	Griewank	$-\frac{1}{4000}\sum_{i=1}^{N_d}(x_i-100)^2+\prod_{i=1}^{N_d}\cos\left(\frac{x_i-100}{\sqrt{i}}\right)-1$	$[-600,600]^{N_d}$	$[0]^{N_d}$	0
f_5	Rastrigin	$-\sum_{i=1}^{N_d}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	$[-5.12,5.12]^{N_d}$	$[0]^{N_d}$	0
f_6	Schwefel	$-418.9829N_d+\sum_{i=1}^{N_d}\left[x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)\right]$	$[-500,500]^{N_d}$	$[420.9687]^{N_d}$	0

πGASR was implemented with the following parameters: crossover probability = 0.8; mutation probability = 0.02; crossover weight factor $w = 0.5$ [40]; mutation shape factor $\beta = 2$ {see [48] for details}. The numbers of generations and chromosomes were 101 and 2500 in Table A2 and 100 and 2000 in Table A3, respectively. In both cases the best chromosome from the previous generation was randomly inserted into the next one ("elitism"). Runs in Table A2 were terminated early if the change in fitness between the current generation and the 20th previous generation was $\leq 10^{-6}$. In order to provide a head-to-head statistical comparison with v-PSO, the πGASR runs in Table A3 ran to completion using all 20,000,000 function evaluations.

Table A2 compares v-PSO and single-run πGASR results for the 10, 20 and 30-dimensional benchmarks. N_d is the DS dimensionality and N_{eval} the total number of πGASR function evaluations. The v-PSO data are average values for 100 runs using 200,000 function evaluations per run (20,000,000 evaluations of each test function). Because v-PSO performs minimization the signs of its results have been changed for comparison to πGASR. In all cases in Table A2, a *single* πGASR run was made because every πGASR run with specific π Fraction distributions yields the same result every time since the fractions are pseudorandom and therefore known with absolute precision.

Table A2. Single Run πGASR Data for v-PSO Benchmark Suite.

fnc#	N_d	Best Fitness		πGASR *Neval*
		v-PSO*	πGASR (single run)	
f_1	10	−1.84e−15 ± 2.9e−16	−5.762878e−4	656,308
	20	−2.84e−15 ± 1.5e−16	−1.161337e−2	328,243
	30	−4.93e−15 ± 3.4e−16	−6.988124e−3	457,978
f_2	10	1 ± 0	0.9999997	457,978
	20	2 ± 0	1.9999993	457,978
	30	3 ± 0	2.9999981	394,558
f_3	10	1 ± 0	0.9999999	361,090
	20	1 ± 3e−18	0.9999999	294,763
	30	1 ± 1e−17	0.9999999	328,243
f_4	10	−0.020 ± 0.006	−0.004429	492,372
	20	−0.0026 ± 0.002	−0.015874	361,090
	30	−8.8568e−4 ± 0.001	−0.002139	457,978
f_5	10	0 ± 0	−1.057361e−4	425,640
	20	0 ± 0	−1.203252e−3	394,558
	30	−5.6843e−16 ± 1e−15	−9.932735e−5	492,372
f_6	10	−620.8131 ± 50.4	−7.753379e−4	457,978
	20	−1.3384e+3 ± 68.5	−7.666976e−4	394,558
	30	−2.1395e+3 ± 103.3	−9.400238e−3	294,763

* average best fitness over 20,000,000 evaluations; data reproduced from Table IV in [49], with sign changed because πGASR maximizes $f(x)$ while v-PSO minimizes.

Table A3. *π*GASR Statistical Data for v-PSO Benchmark Suite.

fnc#	N_d	Average Best Fitness (20 × 10^6 evals)		*π*GASR Overall Best Fitness**
		v-PSO*	*π*GASR Avg / Std Dev	
f_1	10	−1.84e−15 ± 2.9e−16	−3.25606e−3/2.90e−3	−7.22411e−5
	20	−2.84e−15 ± 1.5e−16	−3.71691e−3/4.04e−3	−1.58685e−4
	30	−4.93e−15 ± 3.4e−16	−3.54867e−3/3.03e−3	−1.22941e−4
f_2	10	1 ± 0	0.9999999/2.33e−7	0.9999999
	20	2 ± 0	1.9999997/3.83e−7	1.9999999
	30	3 ± 0	2.9999995/8.41e−7	2.9999999
f_3	10	1 ± 0	0.9999999/9.40e−9	0.9999999
	20	1 ± 3e−18	0.9999999/1.93e−8	0.9999999
	30	1 ± 1e−17	0.9999999/3.55e−8	0.9999999
f_4	10	−0.020 ± 0.006	−5.02147e−4/7.44e−4	−6.39425e−10
	20	−0.0026 ± 0.002	−7.29040e−4/1.25e−3	−3.29534e−09
	30	−8.8568e−4 ± 0.001	−8.45741e−4/1.354e−3	−6.00263e−07
f_5	10	0 ± 0	−2.53755e−4/4.08e−4	−2.73693e−7
	20	0 ± 0	−6.25339e−4/1.20e−3	−1.40063e−6
	30	−5.6843e−16 ± 1e−15	−6.13252e−4/9.69e−4	−1.38614e−7
f_6	10	−620.8131 ± 50.4	−1.84158e−4/8.37e−5	−1.27276e−4
	20	−1.3384e+3 ± 68.5	−4.36412e−4/3.40e−4	−2.55634e−4
	30	−2.1395e+3 ± 103.3	−6.58102e−4/4.18e−4	−3.82090e−4

* average v-PSO best fitness over 20,000,000 evaluations; data reproduced from Table IV in [49], with sign changed because *π*GASR maximizes $f(x)$ while v-PSO minimizes. ** best *π*GASR fitness over 20,000,000 objective function evaluations.

In terms of function evaluations *π*GASR's worst case Figure of 656,308 is nearly 97% less than v-PSO's 20,000,000. In terms of solution quality, *π*GASR performed very well on f_2, f_3 and f_4; well on f_1 and f_5; and exceptionally well on f_6. For f_6 with $N_d = 30$ *π*GASR required 294,763 evaluations, 98.5% fewer than v-PSO, and it returned a best fitness of -9.400238×10^{-3} compared to v-PSO's average value of −2139.5 ± 103.3. These results strongly suggest that pseudorandom *π* fractions can be very useful in implementing what amount to *deterministic* "stochastic" algorithms thereby avoiding the need to make multiple runs to generate statistical data.

Nevertheless, in order to directly compare v-PSO and *π*GASR, Table A3 shows statistical data using the same number of function evaluations. *π*GASR's average best fitness and its standard deviation are tabulated along with the best fitness returned over all runs. *π*GASR performed worse on f_1; essentially the same on f_2 and f_3; better on f_4; worse on f_5; and much better on f_6. However, even in cases where v-PSO outperformed *π*GASR the differences were not dramatic, and for function f_6 πGASR outperformed v-PSO by a very wide margin. Comparing the πGASR data in Table A2 and Table A3 shows that the longer runs with far more function evaluations do yield uniformly better results, as expected.

Summary: π Fractions have been shown to be an effective approach to creating uniformly distributed decision space sample points for global search and optimization. Fractions associated with constants other than π also may be similarly useful, but they have not been investigated. Algorithm πGASR was used an example, and its performance tested against the v-PSO six benchmark suite with generally very good results and in one case much better results. π fraction pseudorandom sequences should be useful for improving the performance of any "stochastic" algorithm in several ways: 1) the resulting sequences are entirely deterministic so that all runs with the same setup produce exactly the same results thus rendering a stochastic algorithm effectively deterministic without compromising its ability to explore the decision space; 2) making successive runs with different sequences likely will result in improved performance with far fewer function evaluations; and 3) decision space adaptation is easily accomplished because the sequences are deterministic (for example, shrinking the decision space around a group of maxima). The π Fraction data file used in πGASR and the source code listings are available online https://github, search term "PiFractions".

Appendix A2. Central Force Optimization

A2.1. The CFO Metaphor

(This material adapted from [43]). Central Force Optimization (CFO) analogizes gravitational kinematics, that is, the motions of real bodies in the real Universe under the influence of real gravity. The governing physical law is Newton's Universal Law of Gravitation. Newton's Law formulates the magnitude of the gravitational force of attraction between the two masses m_1 and m_2 as (see [1] for specific references)

$$F = \gamma \frac{m_1 m_2}{r^2} \tag{a1}$$

where r is the distance between them, and γ is the "gravitational constant". The force of gravity always is attractive, never repulsive, and mass in the real Universe always is positive, never negative. The force of gravity is a *central force* because it acts only along the line connecting the mass centers, hence the name "Central Force Optimization". Mass m_1 experiences a vector acceleration due to mass m_2 given by

$$\vec{a}_1 = -\gamma \frac{m_2 \hat{r}}{r^2} \tag{a2}$$

where $\hat{r}$ is a unit vector that points toward m_1 along the line joining the masses' centers.

A2.2. Problem Statement

The CFO metaheuristic addresses the following problem: In a decision space (DS) defined by $x_i^{\min} \le x_i \le x_i^{\max}, i = 1, \cdots, N_d$ where x_i are the decision variables, locate the global maxima of an objective ("fitness") function $f\left(x_1, x_2, \cdots, x_{N_d}\right)$ possibly subject to a set of constraints $\mathbf{\Omega}$ among the decision variables. The value of $f\left(x_1, x_2, \cdots, x_{N_d}\right)$ is called the "fitness". CFO explores DS by flying metaphorical "probes" whose trajectories are governed by equations of motion drawn from the gravitational kinematics analogy.

A2.3. Constant Acceleration

The location of a mass under constant acceleration is given by the position vector

$$\vec{R}\left(t + \Delta t\right) = \vec{R}_0 + \vec{V}_0 \Delta t + \frac{1}{2} \vec{a} \Delta t^2 \tag{a3}$$

where $\vec{R}\left(t + \Delta t\right)$ is the position at time $t + \Delta t$. $\vec{R}_0$ and $\vec{V}_0$, respectively, are the position and velocity vectors at time t, and the acceleration $\vec{a}$ is constant during the interval Δt. In standard three dimensional Cartesian coordinates the position vector is $\vec{R} = x\hat{i} + y\hat{j} + z\hat{k}$, where $\hat{i}$, $\hat{j}$, $\hat{k}$ are the unit vectors along the x, y, z axes, respectively. The CFO metaphor analogizes Equations (a1)-(a3) by generalizing them to a decision space of N_d dimensions.

A2.4. Probe Trajectory

CFO's probes in a typical three-dimensional DS are shown schematically in Figure A6. The location of each probe at each time step is specified by its position vector $\vec{R}_j^p$, in which p and j are the probe number and time step index, respectively. In an N_d-dimensional DS the position vector is $\vec{R}_j^p = \sum_{k=1}^{N_d} x_k^{p,j} \hat{e}_k$, where the $x_k^{p,j}$ are probe p's coordinates at time step j, and following standard notation $\hat{e}_k$ is the unit vector along the x_k axis.

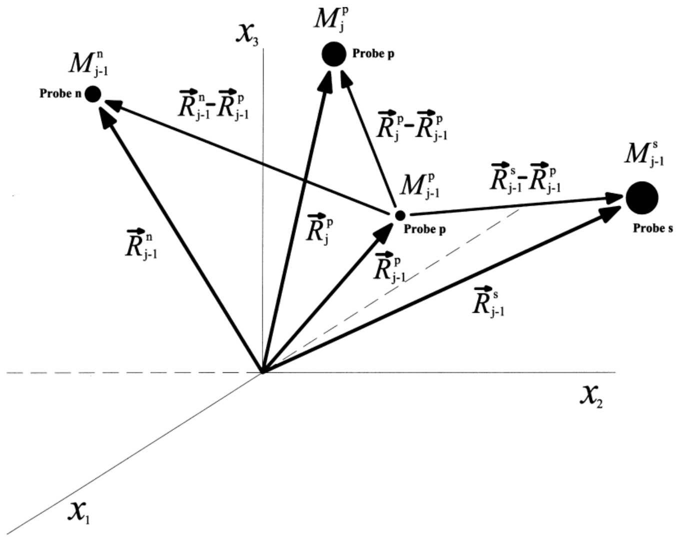


Figure A6. Typical 3-D CFO Decision Space.

Consider a typical probe, p. It moves from position $\vec{R}_{j-1}^p$ at time step $j-1$ to position $\vec{R}_j^p$ at time step j under the influence of the metaphorical "gravitational" forces that act on it. Those forces are created by the fitness at each of the other probes' locations at time step $j-1$. The "time" interval between steps $j-1$ and j is Δt.

At time step $j-1$ at probe p's location the fitness is $M_{j-1}^{p} = f\left(x_1^{p,j-1}, x_2^{p,j-1}, \cdots, x_{N_d}^{p,j-1}\right)$. Each of the other probes also has associated with it a fitness of $M_{j-1}^{k}, k = 1, \cdots, p-1, p+1, \cdots, N_p$, N_p being the total number of probes. In this illustration, the value of the fitness is represented by the size of the blackened circle at the tip of the position vector. In keeping with the gravity metaphor, the blackened circles may be thought of as "planets", say, in our Solar System. Larger circles correspond to greater fitness values, that is, bigger planets with correspondingly greater gravitational attraction. In **Figure A6** the fitnesses ordered from greatest to least occur at $\vec{R}_{j-1}^{s}$, $\vec{R}_{j}^{p}$, $\vec{R}_{j-1}^{n}$, and $\vec{R}_{j-1}^{p}$, respectively, as shown by the relative size of the circles.

Probe p's trajectory in moving from location $\vec{R}_{j-1}^{p}$ to $\vec{R}_{j}^{p}$ is determined by its initial position and by the *total acceleration* produced by the "masses" that are created by the fitnesses (or some function defined on them) at each of the other probes' locations. In the CFO implementation used in this paper the "acceleration", analogous to Equation (a2), experienced by probe p due to the single probe n is given by

$$\frac{G \cdot U\left(M_{j-1}^{n} - M_{j-1}^{p}\right) \cdot \left(M_{j-1}^{n} - M_{j-1}^{p}\right)^{\alpha} \cdot \left(\vec{R}_{j-1}^{n} - \vec{R}_{j-1}^{p}\right)}{\left|\vec{R}_{j-1}^{n} - \vec{R}_{j-1}^{p}\right|^{\beta}} \tag{a4}$$

where G is CFO's "gravitational constant" corresponding to γ in Equation (a1). Note that in the real Universe $G > 0$, always. In CFO space, however, G can be positive (attractive force of gravity) or negative (repulsive force of gravity). Returning to the forces acting on probe p, in a similar fashion to probe n's effect, the acceleration of probe p due to a different probe s is given by

$$\frac{G \cdot U\left(M_{j-1}^{s} - M_{j-1}^{p}\right) \cdot \left(M_{j-1}^{s} - M_{j-1}^{p}\right)^{\alpha} \cdot \left(\vec{R}_{j-1}^{s} - \vec{R}_{j-1}^{p}\right)}{\left|\vec{R}_{j-1}^{s} - \vec{R}_{j-1}^{p}\right|^{\beta}} \tag{a5}$$

Note that the minus sign in Equation (a2) has been included in the order in which the differences are taken in these acceleration expressions. "Mass" in Equation (a2) corresponds to the terms in the numerator involving the fitnesses. Importantly, *it does not correspond to the fitness itself.* In these equations $U(\cdot)$ is the unit step function $U(z) = \begin{cases} 1, & z \geq 0 \\ 0, & \text{otherwise} \end{cases}$. And following standard notation the vertical bars denote vector magnitude, $\left|\vec{X}\right| = \left(\sum_{i=1}^{N_d} x_i^2\right)^{\frac{1}{2}}$, where x_i are the scalar components of $\vec{X}$.

There are no parameters in Equation (a2) corresponding to the "CFO exponents" $\alpha > 0$ and $\beta > 0$, nor to the unit step $U(\cdot)$. In real physical space α and β would take on values of 1 and 3, respectively. Note, too, that the numerators in Equations (a4) and (a5) do not contain a unit vector like Equation (a2). The exponents are included to give the algorithm designer a measure of flexibility by assigning, if desired, a different variation of gravitational acceleration with mass and with distance.

A2.5. Mass in CFO Space

Two other important differences between real gravity and CFO's version are: 1) the definition of "mass", which above is the *difference of fitnesses*, for example, $M_{j-1}^{s} - M_{j-1}^{p}$, not the fitness value itself; and 2) inclusion of the unit step $U(z) = \begin{cases} 1, & z \ge 0 \\ 0, & \text{otherwise} \end{cases}$. The difference of fitnesses is used to avoid excessive gravitational "pull" by other close by probes that presumably will have fitnesses with similar values. The unit step is included to avoid the possibility of "negative" mass. In the physical Universe, mass is positive, always, but in CFO-space the mass could be positive or negative depending on which fitness is greater. The unit step forces CFO to allow only positive masses, that is, attractive masses. If negative fitness differences were allowed, then some accelerations would be repulsive instead of attractive, thus forcing probes away from large fitnesses instead of toward them. The algorithm designer is free to consider other definitions of mass as well. One possibility, for example, might be a ratio of fitnesses similar to the "reduced mass" concept in gravitational kinematics.

A2.6. Total Acceleration and Position Vector for a Single Probe

Taking into account the accelerations produced by each of the other probes on probe p, the *total acceleration* experienced by p as it "flies" from position $\vec{R}_{j-1}^{p}$ to $\vec{R}_{j}^{p}$ is given by the sum of the gravitational effects over all other probes, that is,

$$\vec{a}_{j-1}^{p} = G \sum_{\substack{k=1 \\ k \ne p}}^{N_p} U\left(M_{j-1}^{k} - M_{j-1}^{p}\right) \cdot \left(M_{j-1}^{k} - M_{j-1}^{p}\right)^{\alpha} \times \frac{\vec{R}_{j-1}^{k} - \vec{R}_{j-1}^{p}}{\left|\vec{R}_{j-1}^{k} - \vec{R}_{j-1}^{p}\right|^{\beta}} \tag{a6}$$

Probe p's new position vector at time step j is therefore given by

$$\vec{R}_{j}^{p} = \vec{R}_{j-1}^{p} + \vec{V}_{j-1}^{p} + \frac{1}{2}\vec{a}_{j-1}^{p}\Delta t^{2}, \quad j \ge 1 \tag{a7}$$

where (see discussion)

$$\vec{V}_{j-1}^{p} = 0$$

(a7) is the analog of (a3) where $\vec{V}_{j-1}^{p}$ is the probe's "velocity" at the end of time step $j-1$. In Equation (a7) the coefficient 1/2, the velocity term, and the time increment Δt have been retained primarily as a formalism to highlight the analogy to gravitational kinematics, but they are not required, and in fact $\vec{V}_{j-1}^{p}$ should be set to zero. For the CFO implementation used here, as a matter of convenience Δt is arbitrarily set to 1. Of course, if desired, any constant value of Δt as well as the factor 1/2 can be absorbed into the gravitational constant G. The velocity term $\vec{V}_{j-1}^{p}$ in (a7) has been retained purely as a formality and *should be set to zero* as it is here and shown above. Some tentative numerical experiments showed that including the $\vec{V}_{j-1}^{p}$ term may actually impede probe convergence. The reason for this seemingly contradictory behavior is that while CFO's probe trajectories are piecewise linear, in general they are curvilinear. In curvilinear motion the acceleration and velocity vectors are not necessarily parallel. For example, in the limiting case of circular motion, the velocity vector is tangent to the circle while the acceleration vector is radially inward along the circle's radius, that is, in the case of circular motion the acceleration and velocity vectors are actually orthogonal. In the original CFO paper [1] $\vec{V}_{j-1}^{p}$ serendipitously had been set to zero as a matter of convenience so that the acceleration-velocity directionality issue was avoided entirely.

A2.7. Errant Probes

An important concern is how to handle an "errant" probe, that is, a probe that has flown outside DS. It is possible that the total acceleration experienced by a probe will fly it into regions of unfeasible solutions beyond the DS boundary. There are many ways to deal with this contingency, and a simple one was implemented in the basic version of CFO used here, namely, the use of a "repositioning factor", $0 \le F_{rep} \le 1$. This factor is used to reposition an errant probe according to the formulas

$$\text{If } x_i^{p,j} < x_i^{\min} \therefore x_i^{p,j} = x_i^{\min} + F_{rep} \cdot \left(x_i^{p,j-1} - x_i^{\min} \right) \quad \text{(a8)}$$

$$\text{If } x_i^{p,j} > x_i^{\max} \therefore x_i^{p,j} = x_i^{\max} - F_{rep} \cdot \left(x_i^{\max} - x_i^{p,j-1} \right) \quad \text{(a9)}$$

F_{rep} is assigned an initial value and incremented at each step by a fixed amount ΔF_{rep}, and if it exceeds unity is reset to the initial value. This simple approach guarantees that all probes remain inside DS. Note that while this procedure is inherently pseudo random in nature, for the purpose of improving CFO's exploration of DS numerical experiments have shown that it is not as effective as pseudo randomly injecting a small amount of negative gravity.

A2.8. D_{avg} Convergence Metric

Perhaps the best measure of CFO's convergence is the "Average Distance" metric computed as $D_{avg} = \frac{1}{L_{diag} \cdot \left(N_p - 1 \right)} \sum_{p=1}^{N_p} \sqrt{\sum_{i=1}^{N_d} \left(x_i^{p,j} - x_i^{p^*,j} \right)^2}$, where p^* is the number of the probe with the best fitness, and the superscripts *p* and *j* denote, respectively, the probe and step numbers as above. $L_{diag} = \sqrt{\sum_{i=1}^{N_d} \left(x_i^{\max} - x_i^{\min} \right)^2}$ is the length of the decision space principal diagonal. If every one of CFO's probes has coalesced onto a single point, then $D_{avg} = 0$. How closely this metric approaches zero is a good indicator of how CFO's probe distribution has evolved around a maxima. D_{avg} also is useful in identifying potential local trapping because oscillation in D_{avg} appears to signal trapping at a local maxima [3].

A2.9. Initial Probe Distribution

Every CFO run begins with a *user-specified* Initial Probe Distribution (IPD) defined by two parameters: 1) N_p, the total number of probes used; and 2) where the probes are placed inside DS. Many CFO implementations have employed a *pseudorandom variable* ("prv") IPD comprising an orthogonal array of N_p/N_d probes per axis pseudorandomly deployed on "probe lines" that are parallel to the coordinate axes and intersecting at a point along DS's principal diagonal. *Pseudorandomness* is defined as an arbitrary numerical sequence that is precisely known by specification or by calculation. CFO's fundamentally deterministic nature is not altered by injecting pseudorandomness because at every step CFO's calculations are repeatable with absolute precision (see [3] for a discussion of why pseudorandomness is important in CFO).

Figure A7 provides a two-dimensional (2D) example of this particular type of IPD, in this case nine probes shown on each probe line, two overlapping (but of course any number can be used). The probe lines are parallel to the x_1 and x_2 axes intersecting at a point on DS's principal diagonal marked by position vector $\vec{D} = \vec{X}_{\min} + \gamma\left(\vec{X}_{\max} - \vec{X}_{\min}\right)$, where $\vec{X}_{\min} = \sum_{i=1}^{N_d} x_i^{\min}\hat{e}_i$ and $\vec{X}_{\max} = \sum_{i=1}^{N_d} x_i^{\max}\hat{e}_i$ are the diagonal's endpoint vectors. The parameter $0 \le \gamma \le 1$ [not to be confused with the gravitational constant in Equation (a2)] determines where the probe lines intersect along the diagonal. **Figure A8** shows a typical 2D IPD for different values of γ, and **Figure A9** shows a 3D example using sixteen probes per probe line for a typical "real world" engineering problem, the "Compression Spring". Of course, this procedure is generalized to the N_d-dimensional decision space to create N_d probe lines parallel to the N_d coordinate axes.

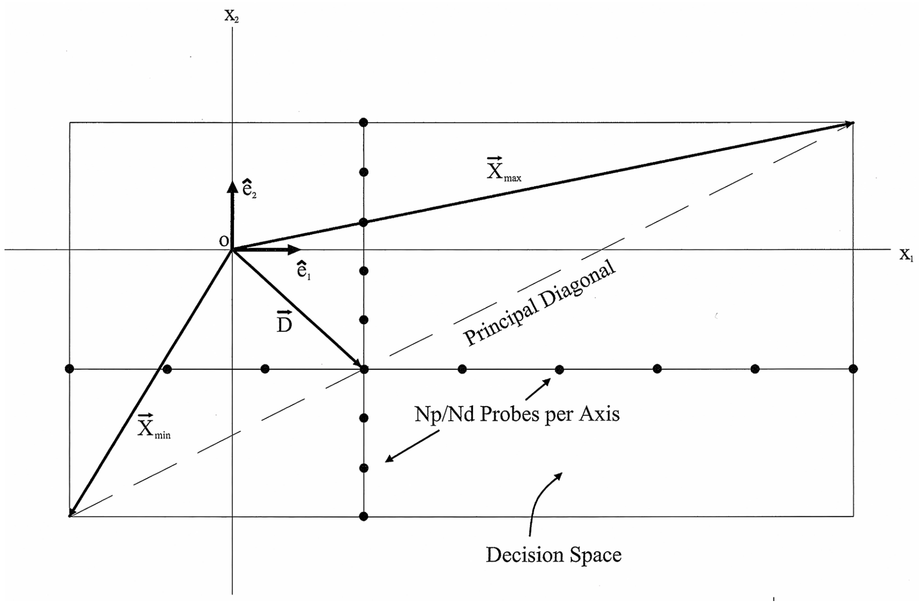
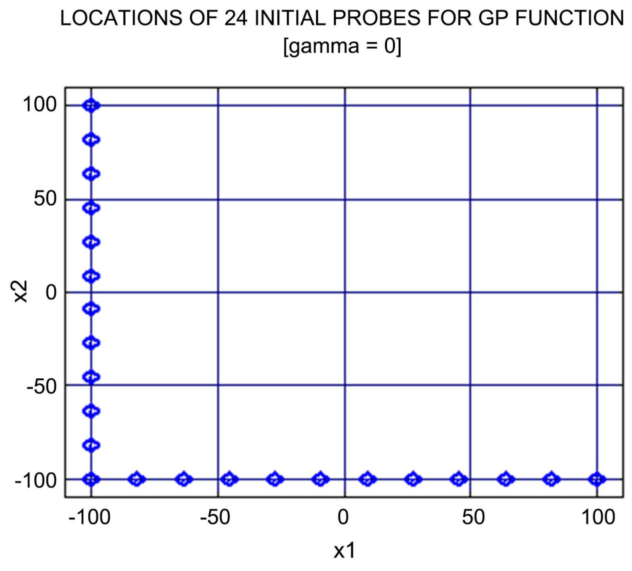
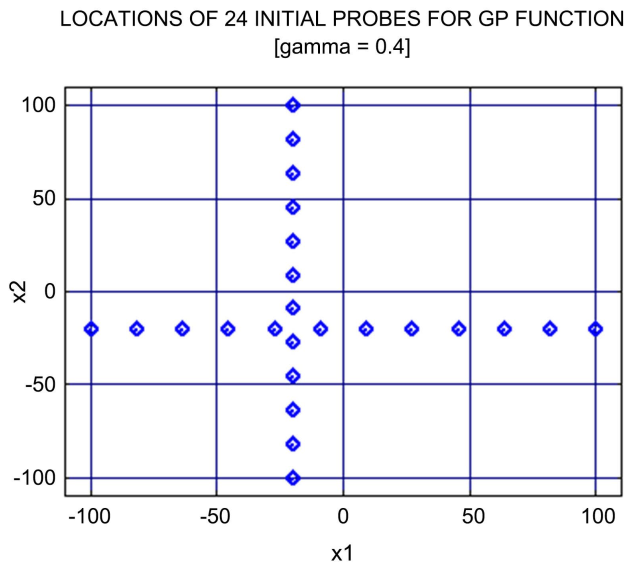


Figure A7. Typical Variable 2D Initial Probe Distribution.

While **Figure A7** shows an equal numbers of probes on each probe line, a different number of probes per axis 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 also might be appropriate if there is any *a priori* knowledge of DS's landscape, however it may have been obtained. For example, denser sampling in one region (more probes) may be appropriate if there appear to be more maxima there. While the variable N_p/N_d IPD of **Figure A7** was used for the results reported here, any number of other altogether different variable IPD's could be used instead. The key idea is that the IPD must be pseudorandom in the sense of uncorrelated with the decision space landscape in order to provide better sampling of the that landscape. Typical CFO pseudocode implementing the variable probe-line IPD approach appears in **Figure A10**.





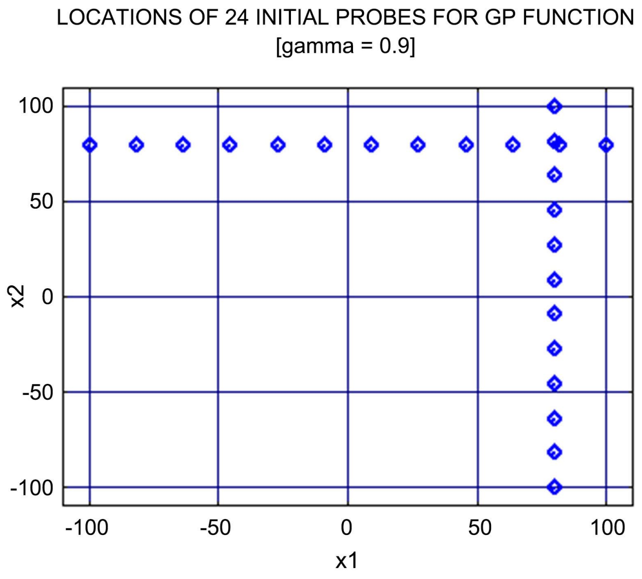


Figure A8. Typical 2D IPD's for Different Values of γ (0./0.4/0.9).

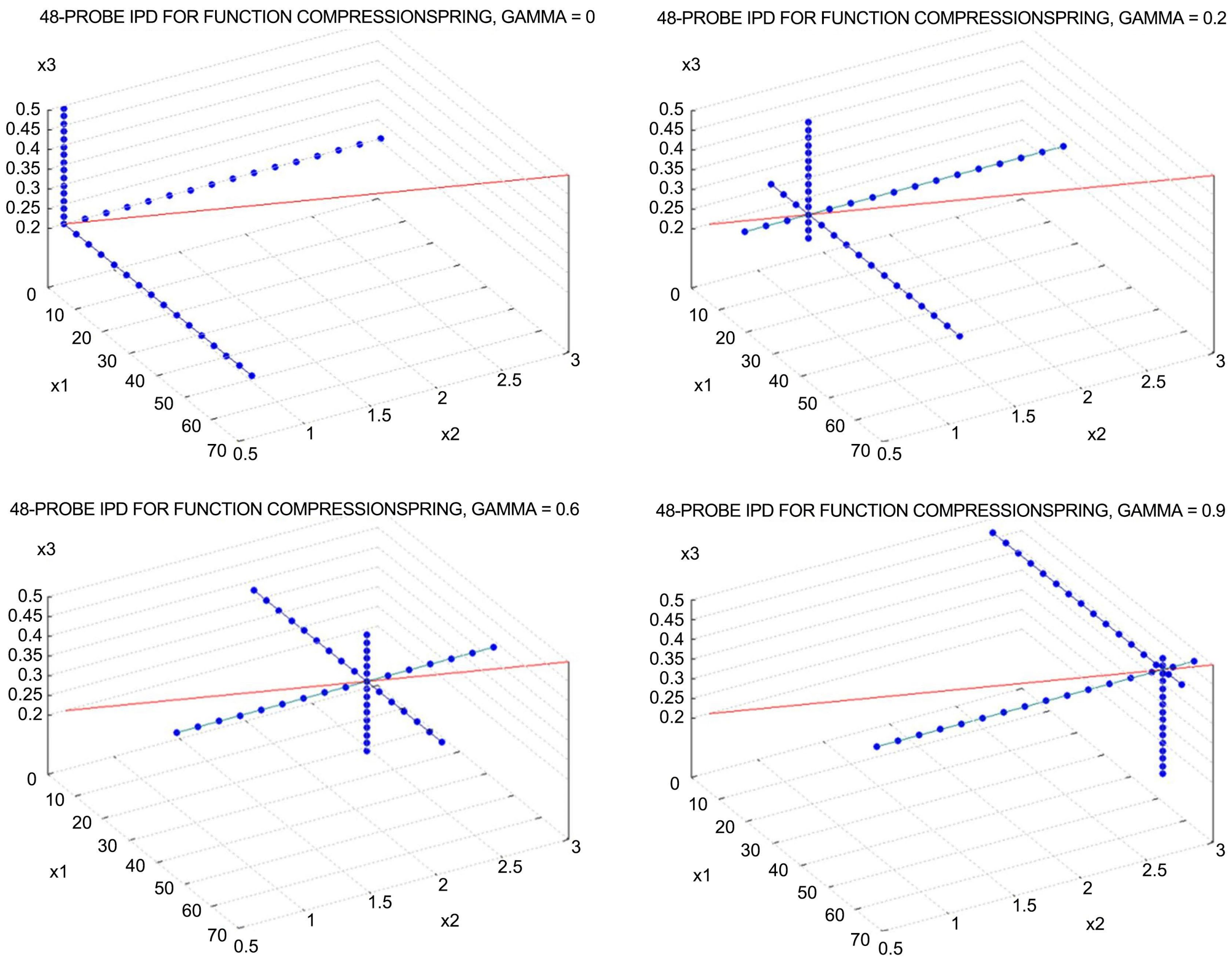
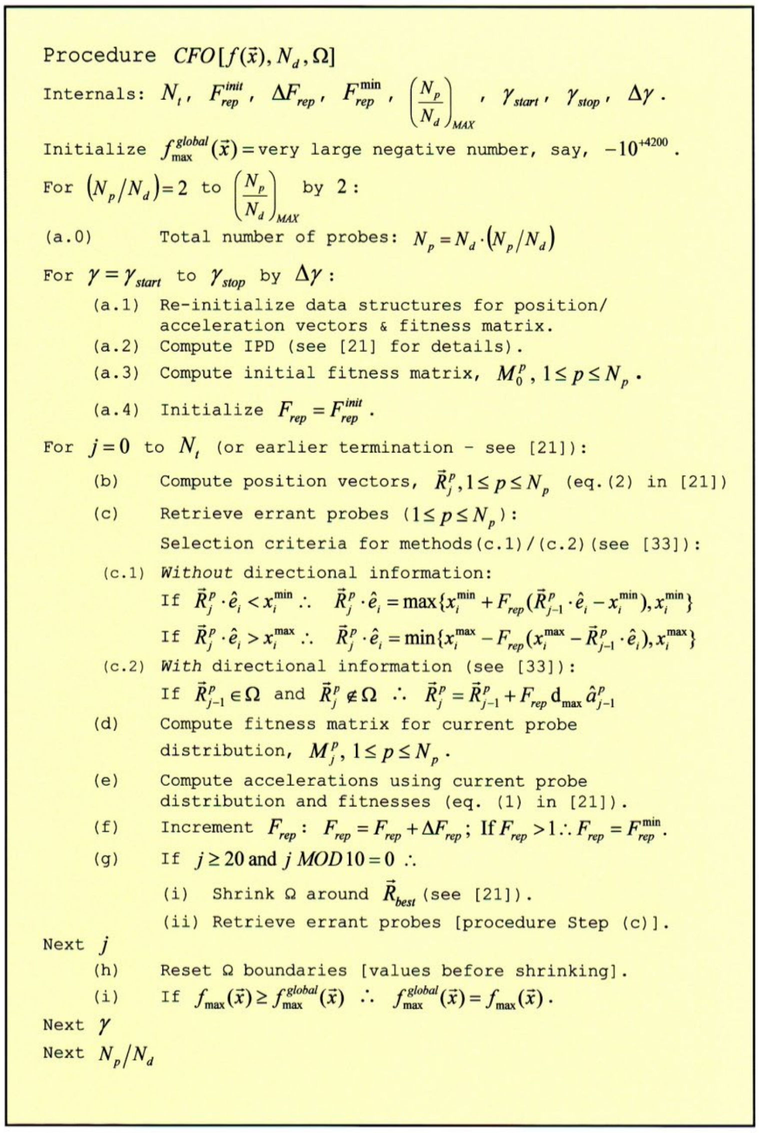


Figure A9. Typical 3D 16-probes/probe line IPD, γ = (0.0/0.2/0.6/0.9, clockwise).



```
Procedure $CFO[f(\vec{x}), N_d, \Omega]$
Internals: $N_t$, $F_{rep}^{init}$, $\Delta F_{rep}$, $F_{rep}^{\min}$, $\left(\frac{N_p}{N_d}\right)_{MAX}$, $\gamma_{start}$, $\gamma_{stop}$, $\Delta\gamma$.
Initialize $f_{\max}^{global}(\vec{x})$ = 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: $N_p = N_d \cdot (N_p/N_d)$
For $\gamma = \gamma_{start}$ to $\gamma_{stop}$ by $\Delta\gamma$:
       (a.1)  Re-initialize data structures for position/
              acceleration vectors & fitness matrix.
       (a.2)  Compute IPD (see [21] for details).
       (a.3)  Compute initial fitness matrix, $M_0^p$, $1 \le p \le N_p$.
       (a.4)  Initialize $F_{rep} = F_{rep}^{init}$.
For $j = 0$ to $N_t$ (or earlier termination - see [21]):
       (b)    Compute position vectors, $\vec{R}_j^p, 1 \le p \le N_p$ (eq.(2) in [21])
       (c)    Retrieve errant probes ($1 \le p \le N_p$):
              Selection criteria for methods(c.1)/(c.2)(see [33]):
        (c.1) Without directional information:
              If $\vec{R}_j^p \cdot \hat{e}_i < x_i^{\min} \therefore \quad \vec{R}_j^p \cdot \hat{e}_i = \max\{x_i^{\min} + F_{rep}(\vec{R}_{j-1}^p \cdot \hat{e}_i - x_i^{\min}), x_i^{\min}\}$
              If $\vec{R}_j^p \cdot \hat{e}_i > x_i^{\max} \therefore \quad \vec{R}_j^p \cdot \hat{e}_i = \min\{x_i^{\max} - F_{rep}(x_i^{\max} - \vec{R}_{j-1}^p \cdot \hat{e}_i), x_i^{\max}\}$
        (c.2) With directional information (see [33]):
              If $\vec{R}_{j-1}^p \in \Omega$ and $\vec{R}_j^p \notin \Omega \ \therefore \ \vec{R}_j^p = \vec{R}_{j-1}^p + F_{rep}\, \mathrm{d}_{\max}\, \hat{a}_{j-1}^p$
       (d)    Compute fitness matrix for current probe
              distribution, $M_j^p$, $1 \le p \le N_p$.
       (e)    Compute accelerations using current probe
              distribution and fitnesses (eq. (1) in [21]).
       (f)    Increment $F_{rep}$: $F_{rep} = F_{rep} + \Delta F_{rep}$; If $F_{rep} > 1 \therefore F_{rep} = F_{rep}^{\min}$.
       (g)    If $j \ge 20$ and $j\ MOD\ 10 = 0 \ \therefore$
              (i)  Shrink Ω around $\vec{R}_{best}$ (see [21]).
              (ii) Retrieve errant probes [procedure Step (c)].
Next $j$
       (h)    Reset Ω boundaries [values before shrinking].
       (i)    If $f_{\max}(\vec{x}) \ge f_{\max}^{global}(\vec{x}) \ \therefore \ f_{\max}^{global}(\vec{x}) = f_{\max}(\vec{x})$.
Next $\gamma$
Next $N_p/N_d$
```

(important note-above, Ω is the Decision Space)

Figure A10. Typical CFO Pseudocode using Probe Line IPD.