A Performance Study on the Effects of Noise and Evaporation in Particle Swarm Optimization

Juan Rada-Vilela   Mengjie Zhang   Winston Seah

School of Engineering and Computer Science
Victoria University of Wellington
PO Box 600, Wellington, New Zealand
{juan.rada-vilela, mengjie.zhang, winston.seah}@ecs.vuw.ac.nz

***Abstract*—This paper presents a performance study on the effects of noise and evaporation in two variants of Particle Swarm Optimization (PSO) on large-scale optimization problems. The variants in consideration are the Synchronous PSO (S-PSO) and the Random Asynchronous PSO (RA-PSO), both of which are evaluated upon the set of benchmark functions presented at the IEEE CEC'2010 Special Session and Competition on Large-Scale Global Optimization. Results show an important detriment to the performance of both variants in the presence of different levels of noise. However, such detriment is significantly mitigated by incorporating an evaporation mechanism into particles to deal with such disruptive effects. Moreover, results show that RA-PSO is significantly better than S-PSO, more tolerant to noise, and better suited for the evaporation mechanism.**

I. Introduction

Noise is an inherent characteristic of real-world problems. Sensor readings and communication channels are examples of cases where information is potentially corrupted by noise. Hence, the design and application of optimization techniques to solve real-world problems demands built-in levels of tolerance to noise by properly handling its disruptive effects.

Particle Swarm Optimization (PSO) is an optimization technique that is sensitive to noise. In PSO, particles explore the search space by iteratively communicating the best solutions found and their respective quality to other particles in the swarm. These interactions enable them to produce new solutions based on their own experience and that of others. However, the quality of such solutions largely depends on how accurate the information is about the problem. That is, if information is subjected to noise, particles will be driven towards solutions whose quality might be significantly worse than that it is expected. Hence, the performance of PSO is compromised in noisy environments.

These circumstances emphasize the importance of incorporating mechanisms into PSO such that particles increase their tolerance to noise. Previous works have proposed to constantly monitor the search space for changes and react to them by partially (or completely) reinitializing the swarm [1, 2], or just doing it periodically [3]. However, monitoring the environment (i.e. reevaluating the solutions) is computationally expensive in large-scale optimization problems, and reinitialization of the swarm is also expensive given the loss of information about the problem and the need to discover new solutions from scratch.

An alternative approach that we consider more reliable to deal with noise in large-scale optimization problems is the evaporation mechanism proposed in [4]. It is inspired by the evaporation of pheromone in Ant Colony Optimization [5], but in this case it iteratively worsens the quality of the best solutions found in order to alleviate the effects of noise with minimal computational cost. This mechanism has been incorporated only into a complex PSO variant based on multiple subswarms, and its performance has been assessed exclusively on the Moving Peak Benchmark function with just five dimensions. This motivates our work to incorporate it into much simpler variants of the PSO algorithm and assess its performance based upon a more challenging set of benchmark functions.

The variants which we are interested in are the Synchronous PSO (S-PSO) and the recently proposed Random Asynchronous PSO (RA-PSO) [6], which is a variant of the Asynchronous PSO (A-PSO) [7]. Previous works have shown that S-PSO and RA-PSO generally produce better results than A-PSO on several unimodal and multimodal functions [6, 8]. However, the S-PSO and RA-PSO algorithms have not yet been compared directly.

The overall objective of this paper is to investigate the effects of noise on the performance of PSO and the ability of an evaporation mechanism to compensate for such effects. Specifically, we focus on:

1) Comparing the quality of the results produced by S-PSO and RA-PSO on large-scale optimization problems under different levels of multiplicative Gaussian noise;
2) Determining the ability of different factors of multiplicative evaporation to serve as an adaptation mechanism for particles in noisy environments; and
3) Analyzing the difference in performance between S-PSO and RA-PSO on large-scale optimization problems.

The rest of this paper is structured as follows. Section II presents the fundamentals of PSO and the variants in consideration, as well as the models of noise in optimization and the evaporation mechanism. Section III details the experimental design to achieve the objectives proposed. Section IV presents the results and their discussion. Finally, in Section V, we present our conclusions and suggestions for further research.

II. Particle Swarm Optimization

PSO was invented by Eberhart and Kennedy in 1995 [9] with inspiration on social models (e.g. bird flocking, fish schooling) and swarming theory. It is a population-based algorithm in which its individuals (known as particles) encode potential solutions to n-dimensional optimization problems. These particles explore the search space through cooperation with other particles by communicating the best solutions found so far and moving towards them.

Particles have a position vector $\mathbf{x}(t)$ and a velocity vector $\mathbf{v}(t)$. The position vector encodes a potential solution to the problem, while the velocity vector is in charge of changing the position accordingly. Thus, the velocity vector balances the trade-off between exploration and exploitation of the search: high velocities result in large changes in the positions of the particles (exploration), whereas low velocities produce small changes (exploitation). Equations 1 and 2 determine the change in position and velocity respectively for particle i in dimension j at iteration $t+1$:

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (1)$$

$$v_{ij}(t+1) = wv_{ij}(t) + c_1 r_1(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_2(t)[\hat{y}_{ij}(t) - x_{ij}(t)] \quad (2)$$

where w is the inertia of the particle [10], c_1 and c_2 are positive acceleration coefficients (a.k.a. cognitive and social components) that weigh the importance of their personal and neighborhood experience, $r_1(t)$ and $r_2(t)$ are random values sampled from independent uniform distributions, $y_{ij}(t)$ is the best position found by particle i in dimension j, and $\hat{y}_{ij}(t)$ is the best position found in dimension j by any particle within the neighborhood.

Communication among particles is based on passing messages containing their own information about the location of the best solutions found and their quality. More concretely, each particle sends a message that contains the best position found and its objective value to other particles that are within the neighborhood. Then, every time a particle receives a message with a position whose objective value is better than that of the previously received messages, the particle stores the position as the best one within its neighborhood.

The topology of the swarm defines the neighborhoods to which each particle communicates its findings. That is, it establishes all the communication links between particles which they can directly communicate with. Several topologies have been proposed in the literature, but the one relevant to this paper is the **star** topology. This topology makes the swarm fully connected, so each message sent by any particle will be received by every other particle within the swarm.

A. Synchronous PSO

The original PSO algorithm [9] was invented with synchronous communications (a.k.a. synchronous updates). That is, particles communicate their findings to their neighbors and obtain in exchange their respective information. Hence, particles have perfect information about their neighborhoods just before updating their positions. Notice that, by perfect information we refer to the case when a particle at iteration t has information about the best position found by each other particle within its neighborhood up to the same iteration. We refer to this algorithm as the Synchronous PSO (S-PSO) and it is described in Algorithm 1.

```
while not stopping condition do
    foreach Particle p ∈ 𝕊 do
        p.evaluate();
        p.communicate();
    end
    foreach Particle p ∈ 𝕊 do
        p.update();
    end
end
```

Algorithm 1: Synchronous PSO (S-PSO).

B. Random Asynchronous PSO

The RA-PSO algorithm was proposed in [6] to model the behavior of the Parallel Asynchronous PSO (PA-PSO) [11, 12] such that results are reproducible. This variant uses asynchronous communications (a.k.a. asynchronous updates) like the A-PSO algorithm, but instead particles are randomly selected following a uniform distribution to evaluate, communicate, and update their positions. Hence, some particles might be selected more than once within the same iteration or might not even be selected at all, thus continuously changing the information flow across iterations and recreating the unbalanced number of operations performed by particles in PA-PSO. Such imbalance causes the swarm to have more diversity as some particles become obsolete from not being selected across iterations, hence potentially driving the swarm away from stagnation in local minima. Additionally, particles have from perfect to different degrees of imperfect information just before updating their positions, thus preventing the loss of diversity in the swarm. This algorithm is described as follows:

```
while not stopping condition do
    for i ← 1 to |𝕊| do
        Particle p ← random(𝕊);
        p.evaluate();
        p.communicate();
        p.update();
    end
end
```

Algorithm 2: Random Asynchronous PSO (RA-PSO).

C. Noise in optimization

There are three types of dynamic environments [3]: **Type I** where the shape of the search space changes but not the location of the optimum, **Type II** where only the location of the optimum changes, and **Type III** where the shape of the search space changes as well as the location of the optimum. For example, consider the search space given by $f(x,t) = x^2$ at time t, then the different types of environments are:

- **Type I**: if $f(x,t+1) = x^4$, then the shape changes but the global minimum remains at $x = 0$.

- **Type II**: if $f(x, t+1) = x^2 + 10$, then the shape remains the same, but it is translated by 10 units.
- **Type III**: if $f(x, t+1) = x^4 + 10$, then the shape changes as well as the location of the global minimum.

Thus, noise can be classified as either type II or III depending on the model used. Additive Gaussian noise, for example, belongs to type II because it is modeled as $f(x, t) = f(x) + N(0, \sigma)$ where $N(0, \sigma)$ is a random value sampled from a Gaussian distribution of mean zero and standard deviation σ. Conversely, multiplicative Gaussian noise belongs to Type III because it is modeled as $f(x, t) = f(x)(1 + N(0, \sigma))$.

Notice that the severity of noise depends on whether it is multiplicative or additive and on its standard deviation σ. The effect of multiplicative noise on the objective function is clearly more disruptive than that of additive noise as the change is larger. However, the effect of the standard deviation is also important since it determines the dispersion of noise. As a reference, consider different values of σ and the 3σ rule (a.k.a. 68-95-99.7 rule) which states that around 68% of the samples from a Gaussian distribution lie within 1σ of the mean, 95% within 2σ, and 99.7% within 3σ.

D. Evaporation in PSO

The idea of having evaporation mechanisms in Swarm Intelligence was first proposed for Ant Colony Optimization to encourage the exploration of new solutions in the search space [5]. This idea was later exploited for PSO in order to make particles tolerant to changes in the search space [4].

The evaporation in particles makes them progressively reduce the objective value of the best position found by themselves and their neighbors. Thus, as iterations pass, particles become less strict by allowing themselves to consider positions with worse objective values than their once personal and neighborhood bests. Hence, particles become tolerant to noise to some extent.

Two evaporation mechanisms have been proposed in [4]: subtractive and multiplicative evaporation. In subtractive evaporation, the objective value of the best personal and neighborhood positions of a particle are subtracted each iteration by a constant amount ρ_s. Conversely, in multiplicative evaporation, the objective values of the best positions are instead multiplied by a factor ρ_m. The subtractive and multiplicative evaporation mechanisms for minimization problems are defined in Equations 3 and 4, respectively,

$$f_i^*(\mathbf{y}_i(t)) = \begin{cases} f(\mathbf{y}_i(t)), & \text{if } f(\mathbf{y}_i(t)) < f_i^*(\mathbf{y}_i(t-1)), \\ f_i^*(\mathbf{y}_i(t-1)) - \rho_s, & \text{otherwise} \end{cases} \quad (3)$$

$$f_i^*(\mathbf{y}_i(t)) = \begin{cases} f(\mathbf{y}_i(t)), & \text{if } f(\mathbf{y}_i(t)) < f_i^*(\mathbf{y}_i(t-1)), \\ f_i^*(\mathbf{y}_i(t-1)) \times (1 + \rho_m), & \text{otherwise} \end{cases} \quad (4)$$

where $\mathbf{y}_i(t)$ and $f_i^*(\mathbf{y}_i(t))$ are the best (personal or neighborhood) position of particle i found until iteration t and its respective objective value, ρ_s is a constant subtractive amount reported to yield good results when $\rho_s \in [-0.9, -0.4]$ (especially when $\rho_s = -0.5$) [4], and $\rho_m \in (0, 1)$ is a constant multiplicative amount reported to yield good results when $\rho_m = -0.989$ [4].

It is important to note that the constant amount of evaporation ρ_s (i.e. subtractive approach) is problem-dependant. For instance, consider the evaporation $\rho_s = -0.5$ (with which they claim the best results were obtained) in a minimization problem which objective function values are within the range $[0, 100]$. In such case, a constant increment of 0.5 is just meaningless, let alone for greater ranges.

Conversely, the multiplicative approach can be used regardless of the problem as long as the range of objective function values is either negative $(-\infty, 0]$ or positive $[0, \infty)$ and the evaporation mechanism is adjusted accordingly. That is, for minimization problems in which the range is positive, the evaporation mechanism is the one defined by Equation 4 because $f_i^*(\mathbf{y}_i(t))$ is worsened by making it tend to infinity. However, when the range is negative, $f_i^*(\mathbf{y}_i(t))$ is instead worsened by making it tend to zero, thus making it necessary to redefine the model as shown in Equation 5.

$$f_i^*(\mathbf{y}_i(t)) = \begin{cases} f(\mathbf{y}_i(t)), & \text{if } f(\mathbf{y}_i(t)) < f_i^*(\mathbf{y}_i(t-1)), \\ f_i^*(\mathbf{y}_i(t-1)) \times \rho_m, & \text{otherwise} \end{cases} \quad (5)$$

E. Related work

Regarding synchronicity of communications in PSO, Rada-Vilela *et al.* [8] compared S-PSO and A-PSO in five unimodal and five multimodal benchmark functions. Their experimentation consisted on 50 independent runs with 300 iterations on each function. The swarms were composed of 30 particles with 30 dimensions each and a ring topology with different numbers of neighbors, namely, $n = \{2, 6, 14, 22, 30\}$. Their observations were based on the results produced by each algorithm in all independent runs and were supported by statistical significance tests using independent samples. They showed that S-PSO performs better than A-PSO in both quality of results and speed of convergence in most benchmark functions.

Rada-Vilela *et al.* [6] compared RA-PSO and A-PSO based on the same benchmarks as in [8] with a similar experimental design. However, in this case the Wilcoxon test had a greater statistical power as it was performed using paired samples instead of independent ones. Their results showed that RA-PSO was significantly better than A-PSO in all unimodal functions, and that it was as good (or even better) in all multimodal functions. Furthermore, its speed of convergence was generally faster than that of A-PSO according to the indicator they used. Hence, they concluded that RA-PSO is a more attractive alternative to the original A-PSO.

Regarding noise, Parsopoulos and Vrahatis [13] analyzed the influence of multiplicative Gaussian noise in S-PSO on three benchnmark functions with two dimensions only. Their results suggest that S-PSO is able to cope with noisy environments. However, we find noteworthy the high rate of success of S-PSO with such high values of standard deviation used to generate the multiplicative Gaussian noise, which is already very disruptive *per se*. The results we find worthy of attention are particularly the success rates (76%, 96% and 80%) on the three problems tackled when $\sigma = 0.9$. Nonetheless, such success might be influenced by the low dimensionality of the functions considered.

For the mechanisms to deal with noise, Fernandez-Marquez and Arcos [4] presented the subtractive and multiplicative evaporation mechanisms described in Section II-D. They incorporated both models into a variant of PSO named multi-Quantum Swarm Optimization, and experimented on the Moving Peaks Benchmark function. They concluded that there is no significant difference between the results using either of the evaporation models.

Later on, Fernandez-Marquez and Arcos [14] presented a dynamic evaporation model based on the velocity of the particles. They argue that a high evaporation factor causes fast adaptation to changes in the search space but the swarm fails to reach good solutions. Conversely, a low evaporation factor causes the swarm to effectively converge but at the cost of a slow adaptation to changes. Therefore, they designed a dynamic mechanism such that the evaporation factor increases when the swarm has converged and a change in the environment has been detected, and decreases when the velocities of the particles are high in order to encourage convergence. However, their mechanism assumes that the optimum objective function value is known *a priori*.

In addition, Eberhart and Shi [3] presented a mechanism in which particles reset their best positions found every 100 iterations in order to avoid stagnation in local minima. Carlisle and Dozier [2] used stationary sentry particles to monitor the search space for changes by reevaluating their positions at different iterations, and if a change in the search space is found, then these particles alert the rest of the swarm to reevaluate their positions. Hu and Eberhart [1] also considered monitoring the search space for changes but used instead a partial or complete reinitialization of the swarm to create diversity and encourage further exploration of the search space.

III. Experimental Design

This section presents the benchmark functions upon which the algorithms are evaluated and the details of the experiments.

A. Benchmark functions

The benchmark functions chosen to measure the performance of the algorithms are those from the CEC'2010 Special Session and Competition on Large-Scale Global Optimization [15]. This suite of benchmarks present different challenges to optimization techniques in terms of the separability of the problem. It comprises 20 minimization functions with different degrees of separability (ranging from separable to fully non-separable) and objective function values within a positive range with a global minimum $f(\mathbf{y}) = 0$. Five groups of functions are defined in this suite as:

- $[F_{01-03}]$ separable, where each dimension can be independently optimized from the others;
- $[F_{19-20}]$ fully non-separable, where any two dimensions cannot be optimized independently;
- $[F_{04-08}]$ partially separable with only a single group of m dimensions that is non-separable;
- $[F_{09-13}]$ partially separable with $\frac{d}{2m}$ groups of m dimensions that are non-separable; and
- $[F_{14-18}]$ partially separable with $\frac{d}{m}$ groups of m dimensions that are non-separable.

The parameters d and m refer to the number of dimensions and the size of the groups, respectively. The default values for such parameters are $d = 1,000$ and $m = 50$, which we also use in our experiments. For further details about these functions, please refer to [15].

B. Experimental setup

We are interested in measuring the performance of the S-PSO and RA-PSO algorithms using different evaporation factors on large-scale optimization problems with different levels of multiplicative Gaussian noise. The evaporation factors were chosen arbitrarily as $\rho_m = \{0.0, 0.05, 0.1, 0.25, 0.5\}$, and the standard deviation values of Gaussian noise were chosen as $\sigma = \{0.0, 0.05, 0.11, 0.22, 0.33\}$ forcing the totality of the samples to lie within 3σ by resampling if needed. Thus, for $\sigma = 0.33$, it is ensured that noise will be at most 1.0 ± 0.99. The goal is that the range of the objective function values remain within the positive range regardless of the amount of noise.

Swarms in both algorithms have 50 particles with 1,000 dimensions each. The communications between them are defined by the star topology, and the acceleration and inertia coefficients are chosen according to the guidelines presented in [16]. In general, we keep a simple configuration of the parameters in order to avoid introducing additional variables that may affect the general performance. Table I summarizes the parameters used.

Table I: Parameter values.

Parameter	Value
Independent runs	50×300 iterations
Number of particles	50 in $\mathbb{R}^{1000}$ with star topology
Evaporation factors	$\rho_m = \{0.0, 0.05, 0.1, 0.25, 0.5\}$
Standard deviation of noise	$\sigma = \{0.0, 0.05, 0.11, 0.22, 0.33\}$
Acceleration	Static with $c_1 = c_2 = 1.49618$
Inertia	Static with $w = 0.729844$
Maximum velocity	$0.25 \cdot \lvert x_{\max} - x_{\min} \rvert$
Velocity clamping	hyperbolic tangent

Experiments are performed as follow. For each σ, a set of five swarms (each with a different evaporation factor) perform 50 independent runs of 300 iterations on each benchmark function. In each run, the five swarms have the exact same initial conditions. That is, a) particles are distributed exactly the same in the search space; b) the seeds for the pseudo-random number generators r_1 and r_2 are different between particles, but the same across swarms; c) the seed for the pseudo-random number generator that selects the next particle is the same across swarms (in RA-PSO only); and d) particles have different seeds for the Gaussian noise generators, but are also the same across swarms. In this way, the results can be compared directly between the same independent runs. That is, results across variants, noise levels, and evaporation factors can be directly compared on a one-to-one basis between each independent run.

Finally, the results we record are made up of the best objective function value found by each swarm on every independent run. Hence, at the end, for each configuration on each benchmark function we record 50 values that correspond to the best solutions achieved in the independent runs. Thus, we refer to the performance of the algorithms in terms of the quality of the results produced.

IV. Results and Discussions

The results produced by RA-PSO and S-PSO are shown in Figures 1 and 2, respectively. These figures show the boxplots that represent the distribution of the objective function values of the best solution produced in every run on each benchmark function. Each subfigure is divided into sections according to the severity of noise, which is identified across the top axis in terms of the standard deviation σ. The bottom axis indicates the different evaporation factors ρ_m: 0.0 (a), 0.05 (b), 0.1 (c), 0.25 (d), and 0.5 (e); while the left axis shows the objective function values. Also, in Figure 1, the boxplots in gray indicate that the results have been scaled down for better visualization, using the following scaling factors:

$$[F_{04}, \sigma = 0.33] : \dot{a} = \frac{1}{2} \qquad [F_{07}, \sigma = 0.33] : \dot{a} = \frac{1}{4}$$

$$[F_{08}, \sigma = 0.22] : \dot{a} = \frac{1}{20} \qquad [F_{08}, \sigma = 0.33] : \dot{a} = \frac{1}{30}$$

$$[F_{08}, \sigma = 0.33] : \dot{b} = \frac{1}{20}$$

We analyze our results according to the quality of the solutions produced when there is: a) absence of evaporation and presence of noise, b) presence of evaporation and absence of noise, c) presence of evaporation and low levels of noise, and d) presence of evaporation and high levels of noise. In each of the following subsections, we begin our discussions with the results obtained with RA-PSO and then contrast them with those obtained with S-PSO.

A. Absence of evaporation and presence of noise

We observe that the performance of both algorithms deteriorates progressively as the presence of noise increases ($\sigma > 0.0$) and no evaporation ($\rho_m = 0.0$) is used to deal with it. In all subfigures, boxplot (a) of each section shows the distribution of the results when no evaporation is used. Notice how the range of the objective function values (left axis) increases significantly as the severity of noise does (top axis).

Such detriment to performance is expected since particles store solutions whose objective function values are corrupted by noise, making it hard (if not impossible) to find other solutions that improve over them. Hence, a whole swarm could be misled into converging to a deceptive solution whose real objective function value is worse than expected.

Also, notice that the performance of S-PSO is significantly worse than that of RA-PSO. We attribute this to the faster speed of convergence of S-PSO by selecting all particles to perform updates in every iteration. Hence, the diversity loss is much greater over time than that of RA-PSO where particles not selected across iterations slow down the convergence and promote diversity to the swarm, thus allowing a better exploration of the search space.

B. Presence of evaporation and absence of noise

Comparing boxplots (a) and (bcde) in the first section of all subfigures, we observe that the performance of both algorithms deteriorates in the presence of evaporation ($\rho_m > 0.0$) and absence of noise ($\sigma = 0.0$). While the range of the objective function values from the results of S-PSO was negatively affected in all functions, RA-PSO was less affected and had instead a few exceptions where evaporation led to even better results. Particularly, RA-PSO produced slight improvements in $F_{07,08}$, and significant ones in $F_{11,16}$.

Such performance is expected since particles have a significant pressure to improve their positions in every iteration in order to avoid the evaporation to lower their expectancy of obtaining better objective function values from their best positions. That is, particles that are selected and do not improve their positions, their expectancy to improve the next time is lowered by the evaporation and hence they will settle with worse positions than with the ones they have already found.

This is particularly detrimental for S-PSO where the expectancy of improvement for the whole swarm largely depends on just one particle holding the best objective function value in each iteration. If such particle is not good enough, the rest of the swarm will most likely evaporate their expectancy in the next iteration and settle with worse solutions. Conversely, when particles in RA-PSO are not selected, the evaporation is not activated and hence particles keep the expectancy of improvement. Moreover, even when they are not selected, they still receive information from the rest of the swarm, thus making them tolerant to iterations where no significant improvements have been found at all.

C. Presence of evaporation and low levels of noise

In RA-PSO, we observe that evaporation values in general are highly benefitial when the severity of noise is low, i.e. $\sigma = \{0.05, 0.11\}$. Notice the quality of the results within the second and third sections of the subfigures where the presence of evaporation, i.e. boxplots (bcde), yields significantly better results than when it is absent, i.e. boxplot (a). This performance is expected since evaporation compensates iteratively for the effects of noise on the objective function values. However, we find interesting the small difference in quality of results between using a large evaporation factor (e.g. $\rho_m = 0.5$) and smaller ones.

On the other hand, the evaporation mechanism in S-PSO did not make any improvement on the results in the presence of low levels of noise. As the evaporation factors increase, the performance of S-PSO significantly deteriorates in most cases with respect to when evaporation is absent ($\rho_m = 0.0$). This can be observed as most of the boxplots (bcde) in the second and third sections are worse than boxplot (a). However, some exceptions in which better results were obtained can be found in $F_{03,04,05,07,11,12,17,19}$ when noise is $\sigma = 0.11$ and the evaporation factor is at least $\rho_m = 0.1$.

D. Presence of evaporation and high levels of noise

Lastly, in RA-PSO we observe that especially large evaporation factors ($\rho_m = \{0.25, 0.5\}$) improve the results when in the presence of high levels of noise ($\sigma = \{0.22, 0.33\}$). Notice in the fourth and fifth sections, the quality increases progressively as the evaporation factors are higher. Again, this performance is expected as the higher the noise the higher the evaporation would be needed to compensate for it.

For S-PSO, the results are generally improved by using evaporation in the presence of high levels of noise, as opposed to what happened with low levels. The most significant improvements can be found in $F_{04,07,08,12,17,19}$ with small evaporation factors, but better results are reported in general with $\rho_m = 0.1$, i.e. boxplot (c).

V. Conclusions and Future Work

In this paper, we have shown the disruptive effects of noise on the performance of the classical S-PSO and the recently proposed RA-PSO in large-scale optimization problems. In both algorithms, noise had a very disruptive effect on the quality of results produced, these being significantly worsened as the level of noise became higher. However, RA-PSO was able to produce significantly better results than S-PSO in all problems regardless of the level of noise.

In order to mitigate the effects of noise on the quality of results, a multiplicative evaporation mechanism has been incorporated into particles such that they iteratively lower their expectations of improving their solutions, thus compensating for the presence of noise in the objective function values of their positions. This mechanism, while not being computationally expensive, makes both algorithms tolerant to noise to some extent. Particularly, its incorporation in RA-PSO always yielded better results than when it was omitted and noise was present. However, such was not the case for S-PSO, which only yielded better results when levels of noise were high and low factors of evaporation were used.

We attribute the superior performance of RA-PSO to the diversity introduced by having particles not being selected across iterations and to the dynamism of potentially having multiple particles as global leaders of the swarm. Both characteristics add a layer of protection against stagnation in local minima. Conversely, the lack of them in S-PSO, causes fast convergence into solutions which leads to a diversity loss and hence a poor exploration of the search space.

This paper can be further extended by:

- Comparing the effects of different models of noise and evaporation factors on different swarm topologies.
- Creating an adaptive evaporation mechanism agnostic to the range of the objective function values.
- Consolidating the performance of RA-PSO by comparing it with other PSO variants in different problems.

References

[1] X. Hu and R. C. Eberhart, "Adaptive particle swarm optimization: Detection and response to dynamic systems," in *IEEE Congress on Evolutionary Computation*, 2002, pp. 1666–1670.

[2] A. Carlisle and G. Dozier, "Tracking changing extrema with adaptive particle swarm optimizer," in *5th Biannual World Automation Congress*, vol. 13, 2002, pp. 265–270.

[3] R. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *IEEE Congress on Evolutionary Computation, 2001*, vol. 1, 2001, pp. 94–100.

[4] J. L. Fernandez-Marquez and J. L. Arcos, "An evaporation mechanism for dynamic and noisy multimodal optimization," in *Genetic and Evolutionary Computation Conference*, 2009, pp. 17–24.

[5] M. Dorigo and T. Stützle, "Ant Colony Optimization," *MIT Press, Cambridge*, 2004.

[6] J. Rada-Vilela, M. Zhang, and W. Seah, "Random Asynchronous PSO," in *5th International Conference on Automation, Robotics and Applications*, 2011, pp. 220–225.

[7] A. Carlisle and G. Dozier, "An off-the-shelf PSO," in *Workshop on Particle Swarm Optimization*, 2001, pp. 1–6.

[8] J. Rada-Vilela, M. Zhang, and W. Seah, "A performance study on synchronous and asynchronous updates in particle swarm optimization," in *Genetic and Evolutionary Computation Conference*, 2011, pp. 21–28.

[9] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *IEEE International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.

[10] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *IEEE World Congress on Computational Intelligence*, 1998, pp. 69–73.

[11] B.-I. Koh, A. George, R. Haftka, and B. Fregly, "Parallel asynchronous particle swarm optimization," *International Journal for Numerical Methods in Engineering*, vol. 67, pp. 578–595, January 2006.

[12] G. Venter and J. Sobieszczanski-Sobieski, "A parallel particle swarm optimization algorithm accelerated by asynchronous evaluations," *Journal of Aerospace Computing, Information, and Communication*, vol. 3, no. 3, pp. 123–137, 2006.

[13] K. E. Parsopoulos and M. N. Vrahatis, "Particle swarm optimizer in noisy and continuously changing environments," in *Artificial Intelligence and Soft Computing*, 2001, pp. 289–294.

[14] J. Fernandez-Marquez and J. Arcos, "Adapting particle swarm optimization in dynamic and noisy environments," in *IEEE Congress on Evolutionary Computation*, 2010, pp. 1–8.

[15] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2009.

[16] F. van den Bergh, "An analysis of particle swarm optimizers," Ph.D. dissertation, University of Pretoria, South Africa, 2002.

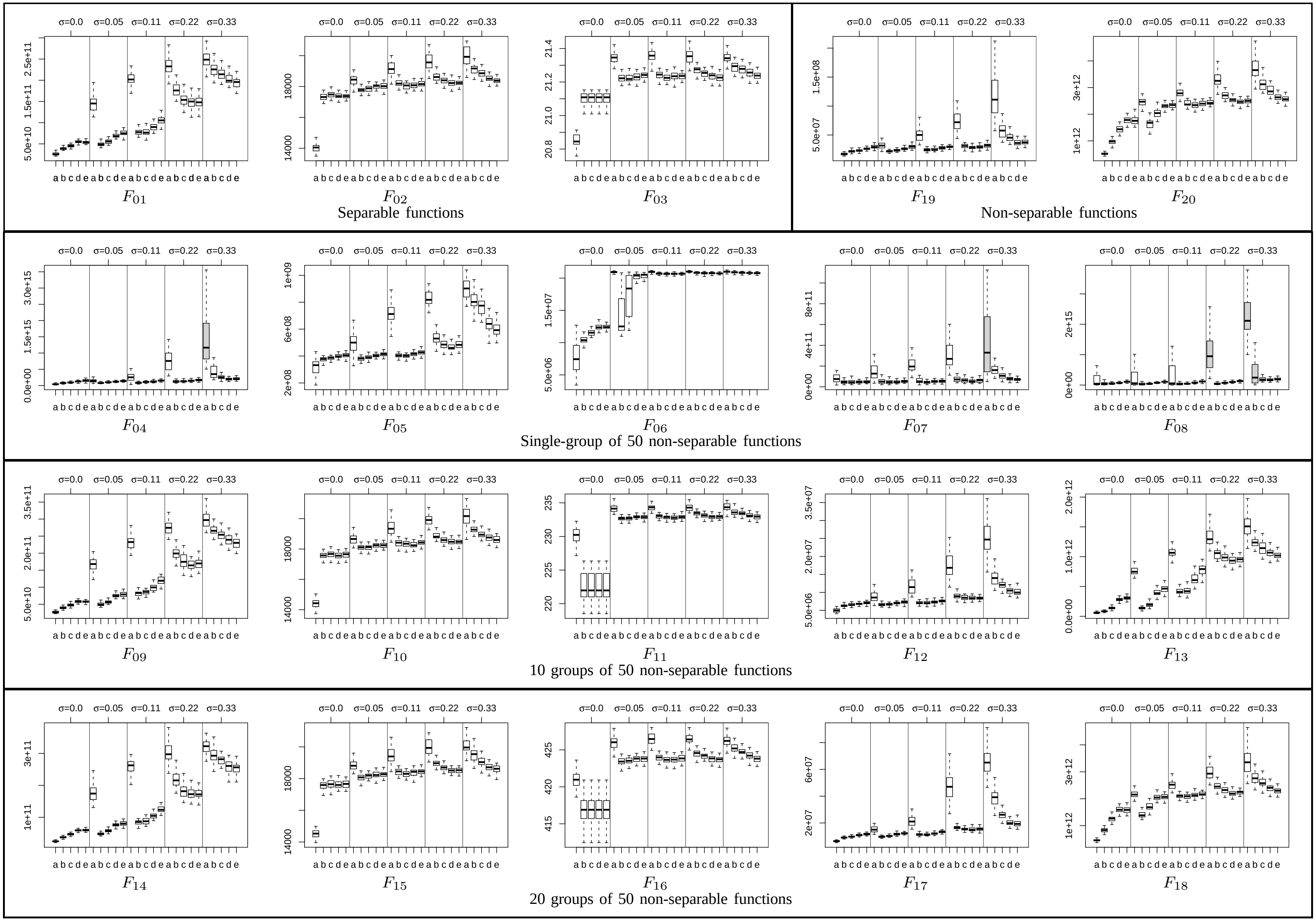

Figure 1: Quality of Results from RA-PSO

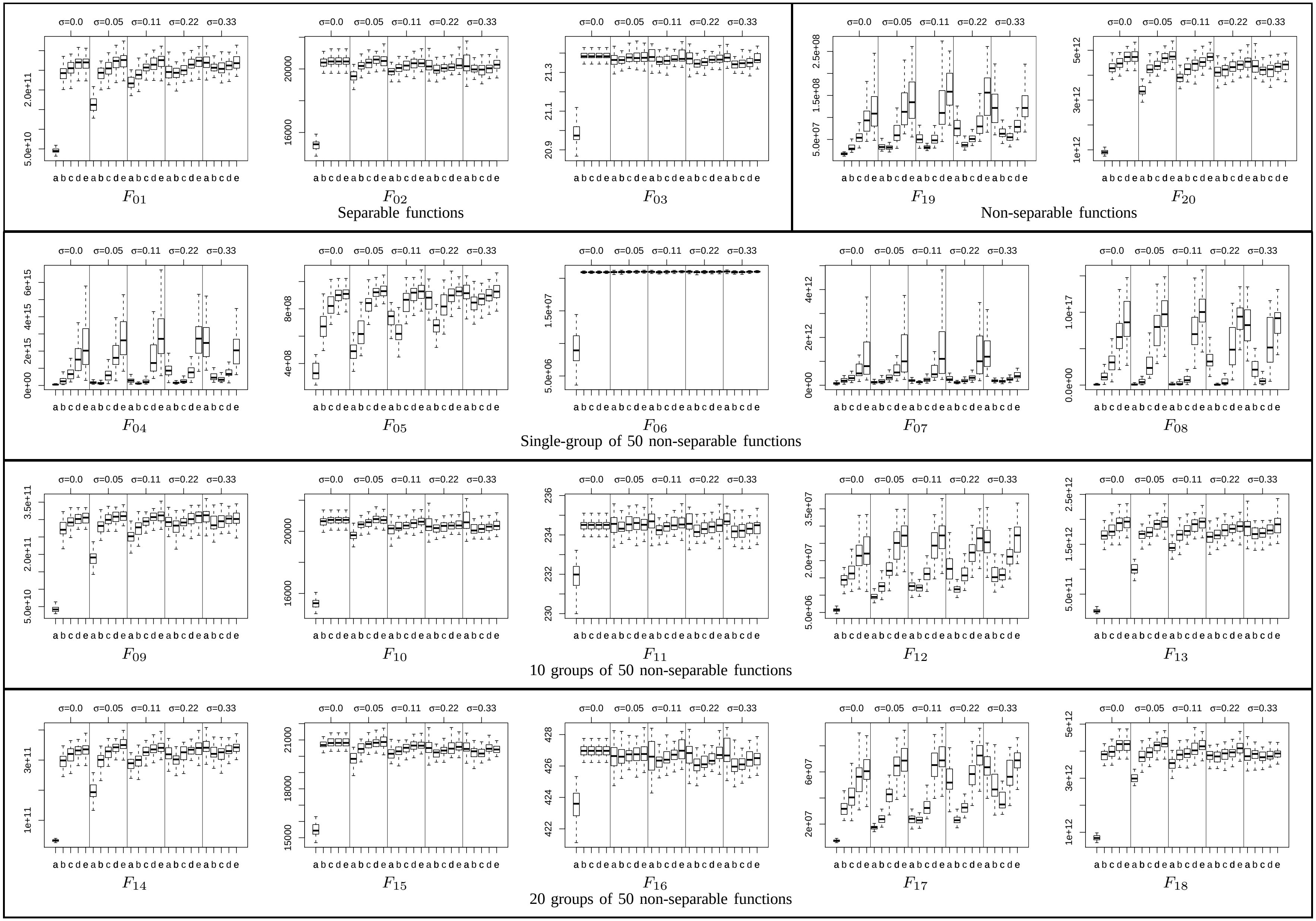

Figure 2: Quality of Results from S-PSO