Heuristic Search Strategies for Noisy Optimization

Manuel Dalcastagné(✉)

Department of Information Engineering and Computer Science, University of Trento, Via Sommarive 9, 38123 Povo, TN, Italy
m.dalcastagne@unitn.it

Abstract. Many real-world optimization problems are subject to noise, and making correct comparisons between candidate solutions is not straightforward. In the literature, various heuristics have been proposed to deal with this problem. Most studies compare evolutionary strategies with algorithms which propose candidate solutions deterministically. This paper compares the efficiency of different randomized heuristic search strategies, and also extends randomized algorithms non based on populations with a statistical analysis technique in order to deal with the presence of noise. Results show that this extension can outperform population-based algorithms, especially with higher levels of noise.

Keywords: Noisy optimization · Heuristics · Statistical analysis

1 Introduction

In many real-world optimization problems, the evaluation of candidate solutions is affected by noise. Possible sources of noise include physical measurement limitations, or the stochastic component employed in simulations. Similarly, in machine learning, the diversity of data used to train and test models adds a layer of uncertainty to the problem. Different models are usually compared using cross-validation approaches, but comparisons are not guaranteed to be correct. In noisy scenarios, since the true value of the objective function is distorted, making correct comparisons between candidate solutions is not straightforward. If the noise is too high with respect to the difference between the true values of two candidates (*signal*), and so the signal-to-noise ratio is too low, comparisons done using a single evaluation per solution might be wrong.

In order to deal with noise, various heuristics have been proposed and studied [2,4,5,17,27]. In particular, many studies employ variants of evolutionary algorithms, which adopt a set of candidate solutions (*population*) subject to local perturbative search and stronger diversification means, often described with terms derived from genetics. Since these algorithms iteratively employ a population to explore the search space and propose new solutions, they are considered to be robust to the presence of noise [2,21]. Multiple works compared various heuristic algorithms with evolutionary strategies, and they have shown

I. S. Kotsireas and P. M. Pardalos (Eds.): LION 14 2020, LNCS 12096, pp. 356–370, 2020.
https://doi.org/10.1007/978-3-030-53552-0_32

that population-based approaches are a good choice to optimize noisy functions [2,4,27]. However, these studies mostly compare evolutionary strategies with algorithms which propose candidate solutions deterministically. But, according to the results of [4], when information about gradients is not available and the objective function is noisy, randomized algorithms might be an effective choice.

Apart from the search policy, which defines how the search space is explored and new solutions are proposed, also other building blocks are necessary to build effective noisy optimization strategies. In fact, to be efficient in this context, algorithms must deal with the presence of noise. In general, this can be achieved in two ways: by increasing the strength of the signal, or by reducing the effect of noise. In randomized algorithms, the signal can be improved by adapting the search region according to the signal-to-noise ratio. Multiple variants of this strategy have been studied in the field of evolutionary algorithms [2,5]. It has been shown that the way in which the search region is adapted during the optimization has a relevant impact. On the other hand, the effect of noise can be reduced by evaluating multiple times each solution [8,9,13,28,29]. Also, if a heuristic is population-based, the impact of noise can be decreased by incrementing the number of candidates (*size*) of the population [3,5,15]. This effect, called *implicit averaging*, has been studied using normally distributed noise with zero mean and different values of constant standard deviations [16,17,22]. Also, [17] shows that increasing indiscriminately the population size can be counterproductive, but without providing an explanation for this behavior. As will be shown empirically, the effect of implicit averaging depends on the type and the amount of noise in the objective function. It is also worth noticing that, in order to deal with noise, randomized algorithms can be extended with statistical analysis techniques. An example is given by simulated annealing [32], which has been extended by adapting the number of samples per solution based on some statistical analysis [1,7]. However, studies which compare diverse randomized algorithms extended with statistical methods are missing in the literature, and this work is a first step in this direction.

This study aims at comparing the efficiency of different heuristic search strategies in the presence of noise, and to investigate the effects that different components of these strategies have on the performance. Differently from previous studies, all the heuristic search strategies employed in this study are randomized, and algorithms not based on populations are extended using a reactive sample size scheme proposed by [14]. The rest of the paper is structured as follows. Section 2 states more formally the noisy optimization problem and gives an overview of the reactive sample size scheme. Section 3 outlines the heuristic search algorithms which have been used in the experiments, and comments the components of the algorithms which are analyzed empirically. Section 4 defines the experiments and analyzes the results.

2 Noisy Optimization

Let F be a stochastic function that models a real world problem. The output of F depends on some decision variables x and on a random vector ξ that represents

the stochasticity of the problem. The expectation of F is defined as

$$f(x) = \mathbb{E}[F(x, \xi)] \tag{1}$$

and it can be estimated by using a sample $\xi_1, ..., \xi_n$ of independent identically distributed (i.i.d.) realizations of the random vector ξ, in order to compute the Sample Average Approximation (SAA) of (1) as

$$\hat{f}_n(x) = \frac{1}{n} \sum_{i=1}^{n} F(x, \xi_i). \tag{2}$$

If the sample $\xi_1, ..., \xi_n$ is i.i.d., by the Law of Large Numbers, as n approaches infinity $\hat{f}_n(x)$ converges to $f(x)$ and so $\hat{f}_n(x)$ is an unbiased estimator of $f(x)$. Moreover, if the variance of F is finite, by the Central Limit Theorem $\hat{f}_n(x)$ asymptotically follows a normal distribution with mean $f(x)$ and variance σ^2/n where σ^2 is the variance of F. As a consequence, the accuracy of the estimation increases with sample size n, but this also increments the computational burden (see also [26,31]). The problem might be defined in the constraints-defined region Θ in which x can assume values as

$$\min_{x \in \Theta} f(x). \tag{3}$$

The SAA defined in (2) can be used as objective function by heuristic optimization techniques, in order to optimize $f(x)$. The presence of noise might require large samples in order to obtain sufficiently accurate estimates, so comparing the performance of different configurations is not straightforward. Given any configuration x, $f(x) - \hat{f}_n(x)$ defines an error $\epsilon_n(x)$ that goes to 0 only in the limit of n going to infinity. As a consequence, when comparing two configurations x_1 and x_2, the difference $\hat{f}_n(x_1) - \hat{f}_n(x_2)$ is not sufficient to decide which configuration has a better average. If the signal $|f(x_1) - f(x_2)|$ is lower than the noise $|\epsilon_n(x_1) - \epsilon_n(x_2)|$, the signal-to-noise ratio is too low and the comparison might be not significant.

2.1 A Reactive Sample Size Algorithm

In this work, in order to deal with the presence of noise, heuristic techniques which do not use a population are extended with a reactive sample size algorithm [14] based on paired t-tests and indifference-zone (IZ) selection. IZ selection is a concept commonly used in ranking and selection (R&S) algorithms. These methods aim at selecting, in a statistically significant manner, the best solution x^* which performs better among a finite set of k possibilities. In R&S methods based on IZ selection, the target is to select the best configuration x^* among a finite set of k configurations, where x^* is better than all other configurations in the set by at least δ and the probability of correct selection (PCS) is $1 - \alpha > 0$, where α is the probability of making an error of type I. δ is called the IZ parameter, and it defines the minimum difference in means considered to be worth detecting. More information about R&S can be found in [10,23,25].

The algorithm works as follows. Given a pair of configurations $\{x_1, x_2\}$ to be compared, paired evaluations are obtained by evaluating x_1 and x_2 using the same ξ_i. Then, these evaluations are used to compute the paired t-test statistic. In fact, as observed by [24], using the same realizations helps to reduce the effect of noise. The correlation among pairs of evaluations reduces the variance with respect to an unpaired statistic. Also, the scheme assumes that F is normally distributed and that its variance is finite.

The algorithm reactively decides, in an online manner, the sample size to be used for each comparison done during the optimization. Also, all the evaluations of solutions previously visited during the search are kept in memory, to avoid the waste of computational budget if a configuration has to be compared multiple times. Significant differences are detected by considering the relationship between probabilities α and β of making an error of type I and type II. To remind the reader, given a null hypothesis H_0 and an alternative hypothesis H_1, α is the probability to reject H_0 when H_0 is true and β is the probability to fail to reject H_0 when H_0 is false. To compute β, a significant difference in means for which H_0 is assumed to be false and H_1 to be true has to be defined. The value for which H_1 is assumed to be true is $\delta_{observed}$, which corresponds to $\hat{f}_n(x_1) - \hat{f}_n(x_2)$. So, given a paired sample, the minimum sample size n that should be used to test a one-tailed hypothesis with error probabilities α and β can be computed. See [14] for more details.

Also, in real world problems, one might not be interested to correctly detect very small differences between means. If x_1 and x_2 have a very similar performance and $\delta_{observed}$ is smaller than a certain user-defined δ, the comparison is done heuristically by considering only the values of $\hat{f}_n(x_1)$ and $\hat{f}_n(x_2)$. The value of δ is expressed as a percentage of the current best solution, because in many cases the user does not know *a priori* the best possible result which can be obtained by the optimization.

3 Optimization Algorithms

The optimization algorithms employed in the experiments are Random Search (RS), the Reactive Affine Shaker (RAS) [11] and the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [19,20]. RS is a simple stochastic local search algorithm which is often used as baseline for comparisons, while RAS and CMA-ES are more advanced stochastic schemes which adapt step size and direction of the search during the optimization.

In RS, a new candidate solution x_{new} is sampled from an interval defined in a neighborhood of the current best solution $x_{current}$, according to a uniform distribution. A step size σ is used to define, as a percentage of the intervals which define Θ along each dimension, the boundaries of the local search region located around $x_{current}$. Consequently, diverse step sizes correspond to search policies with different levels of locality. A step size of 1 would make the search global, and the optimization would correspond to pure random search.

In RAS, a local search region is adapted by an affine transformation. The aim is to scout for local minima in the attraction basin where the initial point

falls. The step size σ and the direction of the search region are adapted in order to maintain heuristically the largest possible movement per function evaluation. The search occurs by generating points in a stochastic manner, with a uniform probability in the search region, following a *double shot* strategy. A single displacement Δ is generated, and two specular points $x_{current} + \Delta$ and $x_{current} - \Delta$ are considered for evaluation. An evaluation is successful if the objective function value in at least one of the two candidates is better than $\hat{f}(x_{current})$. The search region is modified according to the outcome of the comparisons. It is compressed if both comparisons are unsuccessful, and it is expanded otherwise. In both cases, the search region is modified according to an expansion factor $\rho > 1$.

CMA-ES [19,20] is an evolutionary optimization paradigm in which configurations are sampled from a multivariate normal distribution $N(\mu_t, M_t)$, where t defines the iteration of the algorithm. At each iteration, the mean μ_t defines the center of the distribution, the covariance matrix M_t determines shape and orientation of the ellipsoid corresponding to $N(\mu_t, M_t)$, and a step size σ_t controls the spread of the distribution as a percentage of the intervals which define each dimension of Θ. The ellipsoid is the local search region used by the algorithm to explore the search space and propose candidate solutions. Iteratively, CMA-ES follows four steps. First, it samples a fixed number λ of new configurations from $N(\mu_t, M_t)$, creating a population. Second, candidates are evaluated and ranked according to the quality of the evaluations. Third, the best $\lfloor \frac{\lambda}{2} \rfloor$ results are used to update $N(\mu_t, M_t)$, in order to move the search towards the most promising search direction. Fourth, σ_t is increased or decreased according to the length of the so-called evolution paths, in order to maximize the expected improvement of the optimization. This last step is explained more in detail in the following subsection. Also, CMA-ES has been extended with an uncertainty handling (UH) method, to deal with possible noise in the objective function [18]. In this version of CMA-ES, referred as UH-CMA-ES, the uncertainty is measured by rank changes among members of the population. Once each solution of the population has been evaluated and ranked, a few additional evaluations are taken and the population is ranked again. By doing so the algorithm tries to estimate the amount of noise in the evaluations, in order to increase σ_t and prevent the signal-to-noise ratio from becoming too low.

3.1 A Note on CMA-ES Step-Size Adaptation

The adaptation of σ_t, also called cumulative step-size adaptation, is based on the evolution paths mentioned in Sect. 3. An evolution path is a weighted vector sum of the last points successively visited by the algorithm. It provides information about the correlations between points, and it can be used to detect the direction of consecutive steps taken by the optimization. If consecutive steps are going in the same direction (scalar product greater than zero), the same distance could be covered by longer steps and the current path is too long. If consecutive steps are not going in the same direction (scalar product lower than zero), single steps tend to cancel each other out and so the current path is too short. Therefore, to make successive steps more efficient, σ_t is changed accordingly. The step size

determines the signal strength used by CMA-ES to estimate the direction of the gradient. If the steps of the algorithm are very small, the signal is also likely low and therefore the signal-to-noise ratio becomes small as well. Also, it has been shown that in noisy optimization the cumulative step-size adaptation may result in premature convergence [6].

4 Experiments

Before showing the results about the performance of different randomized algorithms in various noisy scenarios and higher dimensions, a preliminary study on CMA-ES is presented. In order to investigate the effects of implicit averaging, the algorithm is tested using populations larger than the standard size proposed by its authors.

4.1 Benchmarking Functions and Noise Models

In order to test diverse heuristic strategies in the presence of noise, Sphere and Rastrigin functions have been extended with multiple types and levels of noise. As in other works in the literature, both functions are optimized in $[-5.12, 5.12]^d$, where d is the number of dimensions. To evaluate the impact of noise on the optimization, a standard practice in the literature is to extend deterministic functions by introducing multiplicative or additive noise. In the case of multiplicative noise, a percentage ϵ of $f(x)$ is added to $f(x)$ according to a displacement generated using a standard normal distribution:

$$f(x, \epsilon) = f(x) + f(x) \cdot \epsilon \cdot N(0, 1). \tag{4}$$

This kind of noise is typical of devices which take physical measurements, like speed cameras, where values are guaranteed to be accurate up to a certain percentage of the measured quantity. However, as the optimization proceeds towards lower values, the noise decreases. This means that, as the optimization approaches the global optimum x^*, it is easier to move into the right direction and, if $f(x^*) = 0$, there is almost no noise in proximity of the global optimum.

Although such a situation is true in many real-world scenarios, there exist other problems where the noise does not always go to zero as the global optimum (if any) is approached. As examples, consider the optimization of simulation models, or the tuning of the hyperparameters of machine learning algorithms. In this case, the noise can be simulated by adopting additive noise. However, determining the amount of noise to add is up to the practitioner. In fact, additive noise is usually normally distributed with zero mean and constant standard deviation σ_ϵ. Since this kind of perturbations does not depend on the signal, the signal-to-noise ratio might cause problems only when approaching the minimum and its effects are going to be very different from function to function.

To avoid these drawbacks, a possibility is to define additive noise as normally distributed with zero mean and dynamic standard deviation. Since the step size

used by randomized algorithms impacts the strength of the signal, in order to test harder noise scenarios it makes sense to set the noise level according to the step size. So, given a point x in Θ and a percentage ϵ, the dynamic standard deviation σ_i along each dimension i is computed as follows. Compute lower bound $l_i = x_i - \epsilon \cdot \Theta_i$ and upper bound $u_i = x_i + \epsilon \cdot \Theta_i$, where Θ_i is the interval in which the function is defined along dimension i. Then, find the minimum m and the maximum M among $\{f(l_i), f(x_i), f(u_i)\}$. Finally, $\sigma_i = |m - M|$ and the additive noise model is defined as

$$f(x, \epsilon) = f(x) + \sum_{i=1}^{N} N(0, \frac{\sigma_i}{k}), \tag{5}$$

where N is the dimensionality of the function and k is a constant used to control the amount of noise. In the experiments, $\epsilon = 0.1$ and $k \in \{1, 2, 3, 6\}$. Therefore, while using this model of noise, the distortion of the signal is set according to the maximum signal which can be detected by an algorithm which adopts a fixed step size (like RS). With $k = 6$, 99.7% of the noise is generated within the intervals which define the local search region. With $k = 3, k = 2, k = 1$ the same is true respectively for $81.86\%, 68, 27\%, 34.14\%$ of the noise.

4.2 Setup

Each experiment is based on 100 macroreplications, where each optimization process has a budget (number of function evaluations) of 5000. Initial solutions are generated according to a uniform distribution defined on the interval of each dimension of Θ. In each experiment, algorithms start the optimization from a randomly generated solution x_0 and consider a local search region of the same size defined around x_0. Then, the local search region is iteratively modified according to the algorithm. Apart from CMA-ES and UH-CMA-ES, the algorithms are employed in two versions: with a naive scheme which uses 1 evaluation for each solution, and the reactive scheme proposed by [14]. The acronym of each algorithm is preceded with N in the former case and with R in the latter. As suggested in [14], the values of the parameters are set as $\alpha_{req} = 0.1$, $\beta_{req} = 0.4$ and $\delta = 0.01$.

In the first set of experiments on CMA-ES, restarts are not considered. In fact, the standard implementation of CMA-ES includes various stopping criterias and restart policies [20], but they have been deactivated in order to improve the analysis of the different components of the algorithm. When activated, all algorithms use global restarts based on a single stopping criterion: if the current best solution does not improve by at least 10% in $k = 500$ function evaluations, a restart is done. An exception is given by RAS, which possibly needs to be restarted because of its double-shot strategy. In fact, if x_0 is generated nearby the boundaries of the search space, the double shot strategy might be unable to generate a valid configuration.

RS uses $\sigma \in \{0.1, 0.2\}$, while RAS employs $\sigma \in \{0.1, 0.2\}$ and $\rho = 2$. CMA-ES and UH-CMA-ES adopt $\sigma \in \{1.0, 2.0\}$, because the library used to implement

the algorithm defines $\sigma \in (0, 10]$[1]. Also, $\lambda = 4 + \lfloor 3 \log d \rfloor$, where d is the dimensionality of the objective function. The values of ρ and λ are the ones suggested respectively by the authors of [11,20].

4.3 Average Loss Signal per Iteration

In a noisy scenario, in order to understand how the algorithm behaves with populations of different sizes, a possible way to proceed is to measure the magnitude of the error made when estimating the gradient. To do so, at each iteration, the population of candidate solutions $p = \{x_1, ..., x_m\}$ is ranked in two ways. Firstly, according to the noisy ranking $\hat{r}$ based on $\hat{f}_1(x_1), ..., \hat{f}_1(x_m)$. Secondly, following the noiseless ranking r defined by $f(x_1), ..., f(x_m)$. Then, the signal loss L is defined as the difference between the signal of these two rankings, where the signal of a ranking is the sum of the absolute differences among the ordered values used for ranking. More formally, in the case of $\hat{r}$ and r,

$$\hat{s}(\hat{r}, p) = \sum_{i=1}^{m-1} |\hat{f}_1(x_i) - \hat{f}_1(x_{i+1})| \tag{6}$$

and

$$s(r, p) = \sum_{i=1}^{m-1} |f(x_i) - f(x_{i+1})|, \tag{7}$$

where the set $\{1, ..., m-1\}$ is ordered respectively according to $\hat{r}$ and r. Therefore,

$$L(\hat{r}, r, p) = \hat{s}(\hat{r}, p) - s(r, p) \tag{8}$$

and the average signal loss per iteration is defined as

$$\mathbb{E}(L) = \frac{1}{N} \sum_{i=1}^{N} L(\hat{r}_i, r_i, p_i), \tag{9}$$

where N is the number of iterations of the algorithm. By following this procedure, the optimization estimates the gradient according to $\hat{r}$ and it is possible to measure by how much the optimization goes in the wrong direction. Also, by comparing $\hat{r}$ and r, the number of misrankings among each population's candidates can be computed, as well as the average misrankings' percentage $\mathbb{E}(M)$ of the whole optimization.

4.4 Results When Using Larger Populations in CMA-ES

This set of experiments is based on the bidimensional Sphere function. As it is possible to see in Table 1, $\mathbb{E}(L)$ keeps increasing as the population grows. Larger populations can potentially detect more signal, but are unable to do so. In fact,

[1] https://github.com/beniz/libcmaes.

Table 1. Performance of CMA-ES with $\sigma = 1.0$ and $\lambda \in \{6, 36, 60\}$ on the Sphere function in 2 dimensions. Macrocolumns show respectively the results obtained when using multiple levels of constant additive noise, multiplicative noise and dynamic additive noise. In each macrocolumn, the average best results $f(x^*)$ are in bold and all values based on $f(x)$ are normalized by the number of dimensions.

		Constant additive				Multiplicative				Dynamic additive			
λ	N	σ_ϵ	$\mathbb{E}(M)$	$\mathbb{E}(L)$	$f(x^*)$	ϵ	$\mathbb{E}(M)$	$\mathbb{E}(L)$	$f(x^*)$	k	$\mathbb{E}(M)$	$\mathbb{E}(L)$	$1f(x^*)$
6	833	0.1	81.92	0.0048	0.02	0.1	13.37	0.0032	**0.00**	6	82.28	0.0392	**0.15**
6	833	1.0	82.55	0.0512	0.22	0.2	24.34	0.0099	**0.00**	3	82.39	0.1219	**1.45**
6	833	2.0	82.74	0.0986	0.42	0.3	33.82	0.0264	**0.28**	2	82.61	0.2362	**3.26**
6	833	3.0	82.85	0.1476	0.75	0.4	43.78	0.0526	**2.29**	1	82.50	1.1282	9.10
36	138	0.1	93.26	0.0138	0.01	0.1	53.48	0.0400	**0.00**	6	95.72	0.1639	0.35
36	138	1.0	95.72	0.1506	0.05	0.2	70.86	0.0662	**0.00**	3	95.85	0.5872	2.26
36	138	2.0	95.98	0.3336	0.11	0.3	79.03	0.1368	0.38	2	95.87	1.3667	4.22
36	138	3.0	96.20	0.4303	0.14	0.4	84.30	0.2789	4.63	1	96.22	4.1927	**9.07**
60	83	0.1	93.75	0.0188	**0.00**	0.1	66.15	0.0508	**0.00**	6	97.10	0.2641	0.40
60	83	1.0	97.05	0.2078	**0.04**	0.2	80.16	0.1322	**0.00**	3	97.25	0.8849	2.49
60	83	2.0	97.50	0.3776	**0.08**	0.3	86.12	0.2235	0.58	2	97.33	2.1270	5.24
60	83	3.0	97.59	0.6161	**0.12**	0.4	90.09	0.4825	6.56	1	97.64	5.7731	10.04

as Figs. 1a, 1b, 1c show, larger populations contribute to mantain a larger step size and so to make wider steps. However, as the amount of noise increases, misrankings are going to happen first among similarly ranked candidates and then also between solutions ranked farther away from each other. Also, the magnitude of the errors increases with the noise. Consequently, larger populations tend to lose more signal and to guide the optimization farther away from the true direction of the gradient.

The effect of the misrankings depends on the amount of noise. Even if $\mathbb{E}(M)$ increases with the population size, it is not implied that the performance of the optimization deteriorates. For example, Table 1 shows that in the case of constant additive noise, larger populations obtain better results. This happens because the signal-to-noise ratio is sufficiently high to avoid misrankings which would guide the optimization towards a significantly wrong direction. The amount of noise starts creating problems only when approaching the minimum, as shown in Fig. 2. With this amount of noise, even N-RS is able to perform comparably well. In contrast, in the case of dynamic additive noise, the signal-to-noise ratio is approximately the same throughout the search space and results are expected to deteriorate much more, as confirmed by the results in Table 1. Even in the case of multiplicative noise, larger populations possibly worsen the performance of the optimization. Therefore, when the signal-to-noise ratio is low and misrankings happen among further positions, increasing the population size or the number of parents is not going to significantly improve the robustness of the optimization.

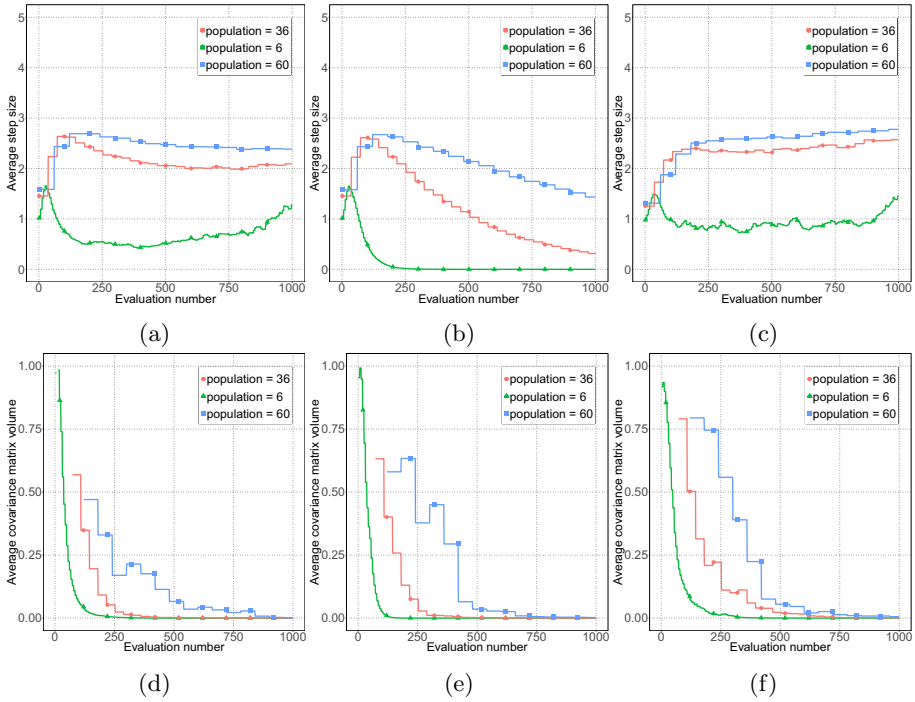


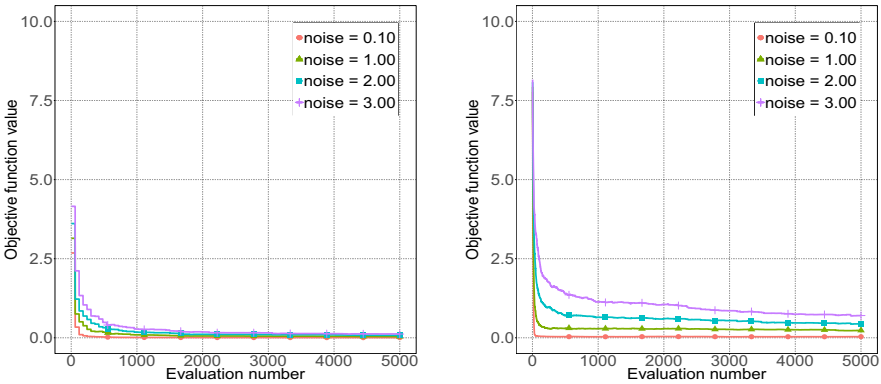
Fig. 1. Each row shows respectively average variation of CMA-ES step size (1a) and CMA-ES covariance matrix volume (1a), with $\sigma = 1.0$ and $\lambda \in \{6, 36, 60\}$. Each column refers respectively to a particular case of the diverse types of noise used in Table 1. More precisely, they show the cases with constant additive noise with $\sigma = 1.0$ (first column), multiplicative noise with $\epsilon = 0.2$ (second column) and dynamic additive noise with $k = 3$ (third column).

In this case, it is preferable to increase the sample size used to estimate each candidate solution.

It is also worth noticing the premature convergence of the covariance matrix. Figures 1d, 1e, 1f show that the volume of the covariance matrix goes to zero in the first part of the optimization. After that, CMA-ES is no longer able to propose significantly different solutions.

4.5 Results When Using Larger Sample Size

These experiments are based on both Sphere and Rastrigin functions, with $d \in \{2, 10\}$. Results in Tables 2 and 3 show that, with high levels of noise, a simple optimization algorithm such as R-RS performs better than more complex algorithms like RAS, CMA-ES or UH-CMA-ES. Without increasing the sample size of estimators, using a population of solutions is not able to compete with single-point algorithms which adapt the sample size of estimators according to empirical evidence. Furthermore, in this context, UH-CMA-ES might even



(a) CMA-ES (b) N-RS

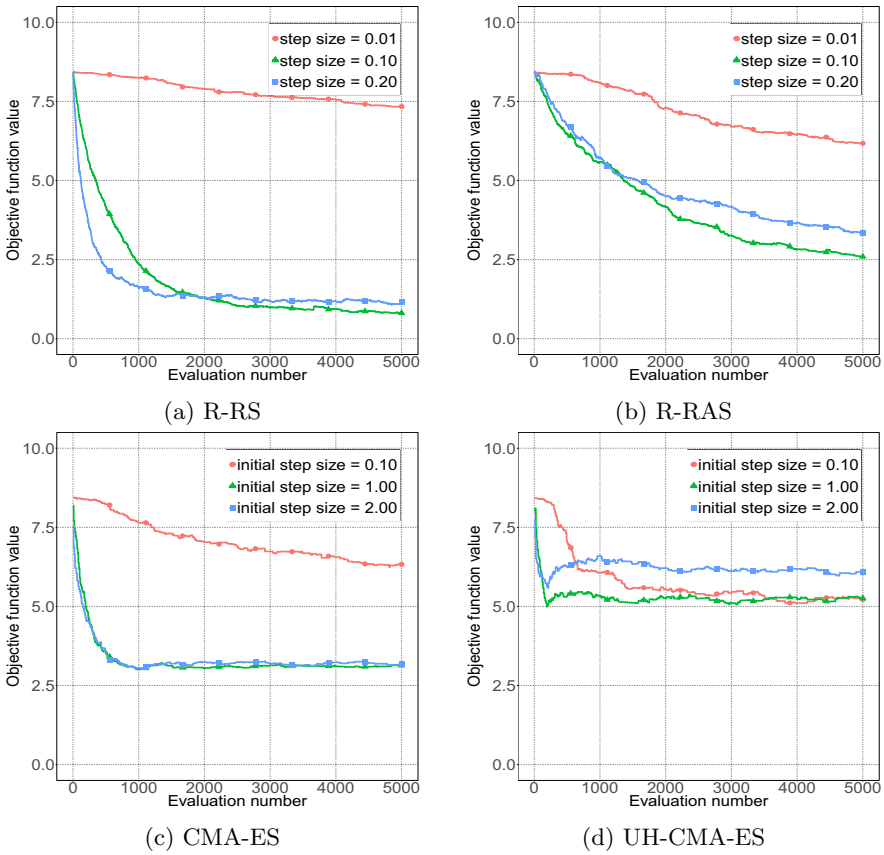
Fig. 2. Average convergence of CMA-ES with $\sigma = 1.0$ and $\lambda = 60$, and N-RS with $\sigma = 0.1$ on the bidimensional Sphere function, with multiple levels of constant additive noise. Lines represent the mean noiseless value of the best configuration found during the optimization (which is not known by the optimizers). Also, all evaluations are normalized by the number of dimensions.

worsen the performance. As shown in Fig. 3d, increasing the step size according to observed misrankings provides better results when the initial step size is very low with respect to the distortion caused by the noise.

Figure 3a shows that a larger step size can improve the efficiency of the optimization. On average, compared solutions correspond to more different estimators and a lower sample size is required to make statistically significant comparisons. However, since a larger step size also implies reducing the sampling granularity, the optimization enhances the global search phase and the convergence speed tends to decrease. For the same reason, as shown in Fig. 3b, the effectiveness of the double-shot strategy in a noisy environment is questionable. Although such an approach can be a good strategy for deterministic optimization, on each iteration very similar configurations are compared, the signal-to-noise ratio tends to be low and the sample size required to make statistically significant comparisons increases. Furthermore, as the search region is compressed, this effect is further enhanced.

In noisy scenarios, step-size adaptation mechanisms and adaptations of the search space are potentially counterproductive. In deterministic functions, compressions of the search region usually lead to a better exploitation of the local structure of the objective function. However, because of the presence of noise, decisions to compress the search region might be wrong and therefore the optimization might prematurely converge to false local optima.

In larger dimensions, the situation changes in the case of Rastrigin function with lower levels of noise. However, it is expected that a population-based algorithm performs better than single-point algorithms in the case of a multimodal function like Rastrigin. Combining a set of candidate solutions at each iteration



(a) R-RS (b) R-RAS

(c) CMA-ES (d) UH-CMA-ES

Fig. 3. Average convergence of the optimizers on the Sphere function, with $d = 10$ $k = 3$. The results obtained by using a very small step size are also shown.

gives the algorithm the ability to adapt to the local topology of the objective function, reducing the risk to get stuck in local optima.

5 Conclusions

This paper investigated different components of diverse heuristic strategies in the context of noisy optimization. A preliminary study on the bidimensional Sphere function showed how the *implicit averaging* effect of population-based algorithms does not always improve the optimization as the size of the population is increased, and analyzed how different amounts of noise change the impact of this effect on the optimization. Randomized algorithms not based on populations have been extended with a statistical analysis technique [14] to deal with the presence of noise, and they have been compared with CMA-ES and UH-CMA-ES.

Table 2. Average best objective function value found by different optimizers, with different levels of dynamic additive noise added to the Sphere function. In each column, the best results are in bold and all values are normalized by the number of dimensions.

		$d = 2$				$d = 10$			
Optimizer	σ	$k = 1$	$k = 2$	$k = 3$	$k = 6$	$k = 1$	$k = 2$	$k = 3$	$k = 6$
R-RS	0.1	**1.14**	**0.29**	**0.15**	**0.05**	4.29	1.75	**0.81**	**0.29**
R-RS	0.2	1.25	0.45	0.26	**0.05**	**3.77**	**1.74**	1.16	0.51
R-RAS	0.1	4.12	1.37	0.78	0.12	6.85	4.27	2.59	0.83
R-RAS	0.2	3.41	1.35	0.55	**0.05**	7.63	4.63	3.34	0.70
CMA-ES	1.0	13.95	5.72	2.45	0.44	11.98	7.22	3.13	0.69
CMA-ES	2.0	14.76	5.00	2.14	0.41	12.17	7.47	3.17	0.71
UH-CMA-ES	1.0	14.71	5.09	2.15	0.45	12.96	9.86	5.26	1.02
UH-CMA-ES	2.0	15.46	5.66	2.03	0.38	12.79	10.33	6.09	0.98

Table 3. Average best objective function value found by different optimizers, with different levels of dynamic additive noise added to the Rastrigin function. In each column, the best results are in bold and all values are normalized by the number of dimensions.

		$d = 2$				$d = 10$			
Optimizer	σ	$k = 1$	$k = 2$	$k = 3$	$k = 6$	$k = 1$	$k = 2$	$k = 3$	$k = 6$
R-RS	0.1	**1.87**	**0.58**	**0.30**	**0.13**	10.94	8.07	7.29	6.60
R-RS	0.2	2.19	0.77	0.48	0.25	**10.30**	**7.24**	6.40	5.93
R-RAS	0.1	7.68	5.65	5.10	3.68	13.20	9.55	8.47	6.97
R-RAS	0.2	6.61	4.77	4.61	2.86	13.98	10.16	8.35	6.88
CMA-ES	1.0	16.85	6.62	3.58	1.03	18.83	12.37	**6.21**	2.24
CMA-ES	2.0	17.48	6.16	2.85	0.98	19.59	13.31	6.46	**2.13**
UH-CMA-ES	1.0	16.79	4.73	2.98	1.11	19.89	15.94	11.02	3.95
UH-CMA-ES	2.0	17.67	5.21	2.11	0.95	19.71	16.57	12.64	3.91

Results in Sect. 4.4 confirm the findings of [17]. The analysis provides an explanation about the reason for which larger populations do not always improve the optimization, and a higher sample size should be preferred when the signal-to-noise ratio is too low. Furthermore, these results also agree with [2]: in the presence of noise, step length control mechanisms are crucial to the performance of the optimization. If optimization methods are extended with statistical analysis techniques such as [14], the resolution at which the search space is explored matters significantly. With lower step sizes, solutions correspond to more similar function evaluations, and the sample size required to statistically determine a difference increases.

Future work aims at extending current results in order to consider other derivative-free optimization strategies and more benchmarks. Also, it would

be worth investigating the performance of more global search policies, which iteratively compare configurations located farther away in Θ and search more locally in different parts of Θ only if sufficient empirical evidence to do so is observed. Approaches like CoRSO [12] or Bayesian Optimization [30] could be good choices.

References

1. Ahmed, M.A., Alkhamis, T.M.: Simulation-based optimization using simulated annealing with ranking and selection. Comput. Oper. Res. **29**(4), 387–402 (2002)
2. Arnold, D.V.: Noisy Optimization with Evolution Strategies, vol. 8. Springer, Cham (2012)
3. Arnold, D.V., Beyer, H.G.: Investigation of the (μ, λ)-es in the presence of noise. In: Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546), vol. 1, pp. 332–339. IEEE (2001)
4. Arnold, D.V., Beyer, H.G.: A comparison of evolution strategies with other direct search methods in the presence of noise. Comput. Optim. Appl. **24**(1), 135–159 (2003)
5. Beyer, H.G.: Evolutionary algorithms in noisy environments: theoretical issues and guidelines for practice. Comput. Methods Appl. Mech. Eng. **186**(2–4), 239–267 (2000)
6. Beyer, H.G., Arnold, D.V.: Qualms regarding the optimality of cumulative path length control in CSA/CMA-evolution strategies. Evol. Comput. **11**(1), 19–28 (2003)
7. Branke, J., Meisel, S., Schmidt, C.: Simulated annealing in the presence of noise. J. Heuristics **14**(6), 627–654 (2008)
8. Branke, J., Schmidt, C.: selection in the presence of noise. In: Cantú-Paz, E., et al. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 766–777. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45105-6_91
9. Branke, J., Schmidt, C., Schmeck, H.: Efficient fitness estimation in noisy environments. In: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, pp. 243–250. Morgan Kaufmann Publishers Inc. (2001)
10. Branke, J., Chick, S.E., Schmidt, C.: Selecting a selection procedure. Manag. Sci. **53**(12), 1916–1932 (2007)
11. Brunato, M., Battiti, R.: RASH: A Self-adaptive Random Search Method, pp. 95–117. Springer, Heidelberg (2008)
12. Brunato, M., Battiti, R.: CoRSO (collaborative reactive search optimization): blending combinatorial and continuous local search. Informatica **27**(2), 299–322 (2016)
13. Cantú-Paz, E.: Adaptive sampling for noisy problems. In: Deb, K. (ed.) GECCO 2004. LNCS, vol. 3102, pp. 947–958. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24854-5_95
14. Dalcastagné, M., Mariello, A., Battiti, R.: Reactive Sample Size for Heuristic Search in Simulation-based Optimization. arXiv e-prints arXiv:2005.12141 (2020)
15. Fitzpatrick, J.M., Grefenstette, J.J.: Genetic algorithms in noisy environments. Mach. Learn. **3**(2–3), 101–120 (1988)
16. Goldberg, D.E., Deb, K., Clark, J.H., et al.: Genetic algorithms, noise, and the sizing of populations. Complex Syst. Champaign **6**, 333–333 (1992)

17. Hammel, U., Bäck, T.: Evolution strategies on noisy functions how to improve convergence properties. In: Davidor, Y., Schwefel, H.-P., Männer, R. (eds.) PPSN 1994. LNCS, vol. 866, pp. 159–168. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58484-6_260
18. Hansen, N., Niederberger, A.S.P., Guzzella, L., Koumoutsakos, P.: A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Trans. Evol. Comput. **13**(1), 180–197 (2009)
19. Hansen, N.: The CMA evolution strategy: a comparing review. In: Lozano, J.A., Larrañaga, P., Inza, I., Bengoetxea, E. (eds.) Towards a New Evolutionary Computation. Studies in Fuzziness and Soft Computing, vol. 192, pp. 75–102. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-32494-1_4
20. Hansen, N.: Benchmarking a bi-population CMA-ES on the BBOB-2009 function testbed. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, pp. 2389–2396. ACM (2009)
21. Hansen, N., Niederberger, A.S., Guzzella, L., Koumoutsakos, P.: A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Trans. Evol. Comput. **13**(1), 180–197 (2008)
22. Harik, G., Cantú-Paz, E., Goldberg, D.E., Miller, B.L.: The gambler's ruin problem, genetic algorithms, and the sizing of populations. Evol. Comput. **7**(3), 231–253 (1999)
23. Hong, L.J., Nelson, B.L., Xu, J.: Discrete Optimization via Simulation, pp. 9–44. Springer, New York (2015)
24. Jian, N., Henderson, S.: An introduction to simulation optimization. In: Yilmaz, L., Chan, W.K.V., Moon, I., Roeder, T.M.K., Macal, C., Rossetti, M.D. (eds.) Proceedings of the 2015 Winter Simulation Conference, pp. 1780–1794 (2015)
25. Kim, S.H., Nelson, B.L.: Chapter 17 selecting the best system. Simul. Handb. Oper. Res. Manag. Sci. **13**, 501–534 (2006)
26. Kim, S., Pasupathy, R., Henderson, S.G.: A Guide to Sample Average Approximation, pp. 207–243. Springer, New York (2015)
27. Nissen, V., Propach, J.: On the robustness of population-based versus point-based optimization in the presence of noise. IEEE Trans. Evol. Comput. **2**(3), 107–119 (1998)
28. Sano, Y., Kita, H.: Optimization of noisy fitness functions by means of genetic algorithms using history of search with test of estimation. In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC 2002 (Cat. No. 02TH8600), vol. 1, pp. 360–365. IEEE (2002)
29. Schmidt, C., Branke, J., Chick, S.E.: Integrating techniques from statistical ranking into evolutionary algorithms. In: Rothlauf, F., et al. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 752–763. Springer, Heidelberg (2006). https://doi.org/10.1007/11732242_73
30. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., De Freitas, N.: Taking the human out of the loop: a review of bayesian optimization. Proc. IEEE **104**(1), 148–175 (2015)
31. Shapiro, A., Dentcheva, D., Ruszczynski, A.: Lectures on Stochastic Programming: Modeling and Theory. 2nd edn. Society for Industrial and Applied Mathematics, Philadelphia (2014)
32. Van Laarhoven, P.J., Aarts, E.H.: Simulated annealing. In: Simulated annealing: Theory and applications, pp. 7–15. Springer, Dordrecht (1987). https://doi.org/10.1007/978-94-015-7744-1_2