The Compact Genetic Algorithm is Efficient Under Extreme Gaussian Noise

Tobias Friedrich, Timo Kötzing, Martin S. Krejca, and Andrew M. Sutton

***Abstract*—Practical optimization problems frequently include uncertainty about the quality measure, for example, due to noisy evaluations. Thus, they do not allow for a straightforward application of traditional optimization techniques. In these settings, randomized search heuristics such as evolutionary algorithms are a popular choice because they are often assumed to exhibit some kind of resistance to noise. Empirical evidence suggests that some algorithms, such as estimation of distribution algorithms (EDAs) are robust against a scaling of the noise intensity, even without resorting to explicit noise-handling techniques such as resampling. In this paper, we want to support such claims with mathematical rigor. We introduce the concept of *graceful scaling* in which the run time of an algorithm scales polynomially with noise intensity. We study a monotone fitness function over binary strings with additive noise taken from a Gaussian distribution. We show that myopic heuristics cannot efficiently optimize the function under arbitrarily intense noise without any explicit noise-handling. Furthermore, we prove that using a population does not help. Finally, we show that a simple EDA called the compact genetic algorithm can overcome the shortsightedness of mutation-only heuristics to scale gracefully with noise. We conjecture that recombinative genetic algorithms also have this property.**

***Index Terms*—Evolutionary algorithms, noisy optimization, run time analysis.**

I. Introduction

EVOLUTIONARY algorithms are widely used for solving real-world optimization problems in uncertain environments. In many practical situations, the evaluation of the objective function is not deterministic, but has a large stochastic component. In these scenarios, evolutionary algorithms must somehow filter the fitness signal from the noise. If the noise intensity is relatively small, this poses little to no problems to selection. However, as the noise intensity grows, the signal becomes more obscured, and the picture is no longer as clear.

In this paper, we want to address the dependence of optimization time on noise intensity (measured as the variance). Specifically, we want to ask whether there is some kind of threshold point in the noise intensity at which the noise becomes too high for efficient optimization, or is it possible for algorithms to be somehow robust to scaling of the noise level?

In the first survey article regarding evolutionary algorithms in noisy environments, Beyer [3] pointed out that the case of *extreme noise* can be easily treated: there is no useful selection information and so an EA will essentially perform a random walk. We prove that this is indeed the case, at least when an algorithm is myopic in the sense that it only makes local changes. However, we also contend that some algorithms, such as estimation of distribution algorithms (EDAs), can leverage the underlying fitness signal in such a way that their behavior never defaults to the diffusion-like behavior observed for myopic algorithms.

To formally characterize how search heuristics can exhibit robustness to noise intensity, we introduce the concept of *graceful scaling* (Definition 1); intuitively, a search heuristic scales gracefully with noise if (polynomially) more noise can be compensated by (polynomially) more resources.

We will consider centered Gaussian noise with variance σ^2 and use OneMax as the underlying fitness function. Already such a seemingly simple setting poses difficulties to the analysis of evolutionary algorithms, as these algorithms are not developed with the analysis in mind. We first prove that there is indeed a noise intensity threshold for myopic algorithms. For simple hillclimbers like randomized local search (RLS) and the $(1+1)$ EA, the threshold is already as low as a constant (Theorem 3). However, we also show that a population cannot help, and the $(\mu+1)$ EA using *any polynomial population size* of at least $\omega(1)$ and a noise intensity of $\sigma^2 = \omega(n^2)$ is provably inefficient (Theorem 4). This implies that the algorithm cannot scale gracefully for Gaussian noise (Corollary 1).

On the other hand, we also investigate a simple EDA known as the compact genetic algorithm (cGA). The working principles of the cGA are in stark contrast to the $(\mu+1)$ EA. Rather than relying on local, myopic mutation operations, the cGA maintains an underlying product distribution that reflect an estimate of the ideal *allele frequencies* a true population would have. In each iteration, it compares two individuals drawn from this product distribution and updates the frequencies based on tournament selection. The only parameter used by the cGA is an update step-size $1/K$ that governs how much the marginal probabilities change in each iteration.

This approach allows the cGA to smooth the noise sufficiently without having to resort to explicit noise-handling

Manuscript received November 19, 2015; revised March 21, 2016 and July 26, 2016; accepted September 13, 2016. Date of publication September 27, 2016; date of current version May 25, 2017. This work was supported in part by the European Union Seventh Framework Programme (FP7/2007-2013) under Grant 618091 (SAGE), and in part by the German Research Foundation under Grant FR 2988 (TOSU).

The authors are with Hasso Plattner Institute, 14482 Potsdam, Germany (e-mail: martin@krejca.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2016.2613739

strategies. We prove that as long as K is sufficiently large, that is $K = \omega(\sigma^2\sqrt{n}\log n)$, then the cGA scales gracefully with Gaussian noise. Specifically, we prove that after $\mathcal{O}(K\sigma^2\sqrt{n}\log Kn)$ many iterations, the cGA will with high probability (w.h.p.) have converged the all-1-string, as desired (Theorem 5) In other words, there is no threshold point in noise intensity at which the algorithm begins to perform poorly.

The proof of Theorem 5 gives insight into how the cGA can filter the signal out of the noise efficiently. Even under intense noise (i.e., a large variance), as long as the allele frequency update $1/K$ is small enough, selection errors due to misclassification by the noisy function are not penalized greatly, and the overall effect of selection biases the stochastic process described by the allele frequencies toward the optimal configuration. This can be contrasted with mutation-only approaches for which such selection errors become fatal in the sense that progress in the correct direction is no longer visible to the algorithm.

We conjecture that evolutionary algorithms that explicitly use recombination can also leverage similar mechanisms to exhibit graceful scaling, however, we do not prove this in the current paper. A step in that direction was recently made by Prügel-Bennett *et al.* [30] who proved that a generational evolutionary algorithm using uniform crossover needs only $\mathcal{O}(n\log^2 n)$ function evaluations to optimize OneMax with additive noise of variance $\sigma^2 = n$.

The remainder of this paper is organized as follows. In Section I-A, we give some background to situate this paper in the broader framework of evolutionary computation in the presence of noise. In Sections II and III, we introduce the algorithms we consider and formalize our settings. In Section IV, we present our main theoretical results. We then also present some numerical experiments in Section V that compare our studied algorithms with a *resampling* approach: a popular noise-handling technique that attenuates the noise level by averaging over repeated samples of the objective function value. We conclude this paper in Section VI.

This paper extends a preliminary version published at ISAAC 2015 [17] in two ways. First, we provide detailed proofs, which were omitted from the extended abstract. Second, we complement our theoretical work with experiments that compare different noise handling techniques.

A. Background

The study of evolutionary algorithms in the presence of noise has already been underway for decades. Fitzpatrick and Grefenstette [16] considered genetic algorithms optimizing a function $f : \{0,1\}^n \to \mathbb{R}$. However, for any string x, instead of direct access to a fitness value $f(x)$, the GA is only able to perform a fixed number of samples of a random variable with mean $f(x)$. They explored the tradeoff between the population size and this fixed number of samples. They argued that the implicit parallelism of a GA is a sufficient mechanism for handling noise and that in some cases, the amount of explicit sampling can be reduced by increasing the population size. However, it remains an open question exactly how this tradeoff between explicit resampling and implicit sampling with a population works.

Beyer [3] presented a survey on evolutionary algorithms in the presence of noise, focusing on continuous optimization. Among other things, he identifies three measures to improve the performance of EAs on noisy functions: 1) resampling; 2) population sizing; and 3) inheriting rescaled mutations.

A first categorization of different types of noise in optimization was given by Beyer *et al.* [6] and further developed in [4]. The authors delineate several types of noise: 1) noise from the environment; 2) actuator imprecision (noise in the decision variables); 3) imprecision of the evaluation of system output; and 4) uncertainties regarding feasibility constraints. In this paper, we focus on uncertainty of type 3) in which the measurement of the objective function is perturbed by some additive stochastic noise term. This noise model has also been called *additive posterior noise* in [18] and has been studied in the context of combinatorial optimization in several recent papers [10], [15], [19], [33]. In a recent survey of Jin and Branke [23], the additive noise model is simply called *noisy fitness*. In this survey, the authors compare the noisy fitness model to the problem of finding *robust solutions*, that is, solutions that perform best after the design variables somehow change after optimization. They also consider time-varying fitness functions, which are deterministic at any point in time, but change as a function of time.

In the context of discrete optimization, Gutjahr and Pflug [19] extended the classical convergence result for simulated annealing to objective functions perturbed by an additive zero-mean noise term. They show that if the variance of the noise is reduced quickly enough (e.g., by resampling) then the algorithm converges for suitable cooling schedules.

The rigorous analysis of the runtime of evolutionary algorithms on noisy functions on discrete domains was initiated in 2004 by Droste [13]. In that paper, a noisy variant of the OneMax test function was analyzed for the simplest EA, the $(1+1)$ EA. In essence, it was shown that the $(1+1)$ EA can deal with small noise levels, but not medium noise levels.

A common technique for coping with noise is *resampling*. In this strategy, a noisy objective function is measured multiple times at a given point and the average value is computed. This has the effect of explicitly reducing the variance of the noise. However, resampling comes at the extra cost of potentially many more function evaluations, and the study of the tradeoff between noise reduction and computational effort of resampling has been investigated for many applications [2], [5], [7], [8]

Recently, Akimoto *et al.* [1] explicitly studied the effect of resampling on various noise models to derive the extra cost incurred by performing enough resampling to ensure the underlying optimization algorithm sees a noiseless function. For Gaussian noise, they show the existence of a resampling scheme such that any optimization algorithm that requires $r(\delta)$ function evaluations to optimize a noise-free function f with probability $1-\delta$ requires $\max\{1, 32\sigma^2(\ln(2) - \ln(1-(1-\delta)^{1/r(\delta)}))\}$ evaluations to optimize $f + \mathcal{N}(0,\sigma^2)$ with probability $(1-\delta)^2$ under their resampling scheme. Note that this

Algorithm 1: RLS Optimizing f

```
Choose x ∈ {0,1}^n u.a.r.;
while termination criterion not met do
    Create y by flipping exactly one bit in x u.a.r.;
    if f(y) ≥ f(x) then x ← y
```

can be upper-bounded by $32\sigma^2(\ln(2)+1) < 20e\sigma^2$, assuming $\lim_{\delta\to 0}(1-\delta)^{1/r(\delta)} = 0$.

In the next part of this paper, we want to analyze the run time of algorithms without any explicit noise handling techniques. However, we will revisit the question of resampling in the experiments in Section V.

II. Algorithms

We consider optimization of *pseudo-Boolean* fitness functions, that is, of functions $f : \{0,1\}^n \to \mathbb{R}$ (where n is the fixed problem dimension). Thus, our search space is $\{0,1\}^n$, and the individuals of population-based algorithms will be taken from that set. For any $x \in \{0,1\}^n$, let $|x|_0$ denote the number of 0s of x, and let $|x|_1$ denote the number of 1s of x. We will consider randomized fitness functions as a model of noise. Thus, two applications of a fitness function f can lead to different results. In all algorithms, each search point is evaluated for its fitness value in each iteration anew, after each iteration the fitness evaluation is discarded, even if the search point is not.

A. Simple Hillclimbers

The simplest randomized search heuristics maintain a single current solution and iteratively apply some search operator to generate a neighboring point. If the neighboring point is at least as fit as the current point, it is accepted as the new current solution. In RLS (Algorithm 1) a neighboring point is produced by flipping exactly one bit chosen uniformly at random. This is a *local* mutation operation, meaning that only Hamming neighbors of a solution x are created. A type of *global* mutation is obtained by flipping each bit independently with probability $1/n$. In this way, any bitstring is created with nonzero probability, however, the probability decays exponentially with Hamming distance from x. Implementing a hillclimber with such a global mutation operation results in the $(1+1)$ EA (see Algorithm 2 with $\mu = 1$).

B. Population-Based Mutation-Only EA

The $(\mu+1)$ EA, defined in Algorithm 2, is a simple mutation-only evolutionary algorithm that maintains a population of μ solutions and uses elitist survival selection. It derives its name from maintaining a population of μ individuals (randomly initialized) and generating one new individual each iteration by mutating a parent chosen uniformly at random from the current population. Then it evaluates the fitness of all individuals and chooses one with minimal value to be removed from the population, so that again μ individuals proceed to the next generation.

Algorithm 2: $(\mu+1)$ EA Optimizing f

```
t ← 0;
P_t ← μ elements of {0,1}^n u.a.r.;
while termination criterion not met do
    Select x ∈ P_t u.a.r.;
    Create y by flipping each bit of x independently with
    probability 1/n;
    Let z ∈ P_t ∪ {y} chosen s.t. ∀v ∈ P_t ∪ {y}: f(z) ≤ f(v);
    P_{t+1} ← P_t ∪ {y} \ {z};
    t ← t + 1;
```

Algorithm 3: cGA Optimizing f

```
t ← 0;
p_{1,t} ← p_{2,t} ← ··· ← p_{n,t} ← 1/2;
while termination criterion not met do
    for i ∈ {1,...,n} do
        x_i ← 1 with probability p_{i,t}, x_i ← 0 with
        probability 1 − p_{i,t};
    for i ∈ {1,...,n} do
        y_i ← 1 with probability p_{i,t}, y_i ← 0 with
        probability 1 − p_{i,t};
    if f(x) < f(y) then swap x and y for i ∈ {1,...,n} do
        if x_i > y_i then p_{i,t+1} ← p_{i,t} + 1/K if x_i < y_i then
        p_{i,t+1} ← p_{i,t} − 1/K if x_i = y_i then p_{i,t+1} ← p_{i,t}
    t ← t + 1;
```

C. Estimation of Distribution Algorithms

EDAs build and sample explicit probability distributions to optimize functions. This is contrasted with traditional evolutionary algorithms that often sample implicitly from a probability distribution by way of different evolutionary operators.

One of the simplest EDAs is the cGA [21]. The cGA derives its name from the fact that it maintains a population of size K *implicitly* in memory. Rather than storing each individual separately, the cGA only keeps track of population *allele frequencies* and updates these frequencies during evolution. That is, instead of a population $P \subseteq \{0,1\}^n$ with K individuals, the cGA stores a vector $p = (p_1, \ldots, p_n)$, where

$$p_i = \frac{1}{K}\sum_{x\in P} x_i.$$

Thus, p_i represents the frequency of 1s in position i, and at any point in time the internal state of the cGA is completely characterized by this frequency vector p. Offspring are then generated according to these allele frequencies.

The first rigorous analysis of the cGA is due to Droste [14], who gave a general run time lower bound for all pseudo-Boolean functions, and a general upper bound for all linear pseudo-Boolean functions. Defined in Algorithm 3, the cGA maintains for all times $t \in \mathbb{N}_0$ a frequency vector $(p_{1,t}, p_{2,t}, \ldots, p_{n,t}) \in [0,1]^n$. In the tth iteration, two strings x and y are sampled independently from this distribution, where

Algorithm 4: Noise-Oblivious Scheme for A

```
i ← 0;
repeat until solution found
    Run A(2^i) for T_δ(2^i) steps and stop it afterward;
    i ← i + 1;
```

$\Pr(x = z) = \Pr(y = z) = (\prod_{i:z_i=1} p_{i,t}) \times (\prod_{i:z_i=0}(1 - p_{i,t}))$ for all $z \in \{0, 1\}^n$. The cGA then compares the objective values of x and y, and updates the distribution by advancing $p_{i,t}$ toward the component of the winning string by an additive term of $1/K$. This allele frequency update is inspired by the dynamics of a concrete population of size K undergoing steady-state binary tournament selection: in that case the proportion of each winning allele also increases by exactly $1/K$ [21]. We will assume without further mention that there is an integer k such that $k/K = 1/2$; this implies that, at all times t and for all $i \leq n$

$$p_{i,t} \in \left\{0, \frac{1}{K}, \ldots, \frac{1}{2} - \frac{1}{K}, \frac{1}{2}, \frac{1}{2} + \frac{1}{K}, \ldots, 1 - \frac{1}{K}, 1\right\}.$$

D. Noisy Fitness

Let F be a family of pseudo-Boolean functions $(F_n)_{n\in\mathbb{N}}$, where each F_n is a set of (unnoisy) functions $f : \{0, 1\}^n \to \mathbb{R}$. Let D be a family of distributions $(D_\theta)_{\theta\in\mathbb{R}}$ such that for all $D_\theta \in D$, $\mathrm{E}(D_\theta) = 0$. We define F with additive D-noise as the set $F[D] := \{f_n + D_\theta : f_n \in F_n, D_\theta \in D\}$.

Definition 1: An algorithm A *scales gracefully with noise* on $F[D]$ if there is a polynomial q such that, for all $g_{n,\theta} = f_n + D_\theta \in F[D]$, there exists a parameter setting p such that $A(p)$ finds the optimum of f_n using at most $q(n, \theta)$ calls to $g_{n,\theta}$.

Algorithms that operate in the presence of noise often depend on *a priori* knowledge of the noise intensity (measured by the variance). In such cases, the following scheme can always be used to transform such algorithms into one that has no knowledge of the noise intensity. Given an algorithm $A(\sigma^2)$, let $T_\delta(\sigma^2)$ denote the number of steps it takes A to solve a noisy function with variance at most σ^2 with probability at least $1 - \delta$. A *noise-oblivious scheme* for A is in Algorithm 4. This algorithm embodies a simple restart strategy.

If an algorithm A scales gracefully with noise, then the noise oblivious scheme for A scales gracefully with noise. The following proposition holds by a simple inductive argument.

Proposition 1: Suppose $f_{n,v} \in F[D]$ is a noisy function with unknown variance v. Fix n and assume that, for all $c > 0$ and all x, $cT_\delta(x) \leq T_\delta(cx)$. Then for any $s \in \mathbb{N}^+$, the noise-oblivious scheme optimizes $f_{n,v}$ in at most $T_\delta(2^s v)$ steps with probability at least $1 - \delta^s$.

Proof: By the assumptions on T_δ, for all c, x, $cT_\delta(x) \leq T_\delta(cx)$ and so by induction, for any $k \in \mathbb{N}$, $\sum_{i=0}^{k} T_\delta(2^i) \leq T_\delta(2^{k+1})$. Let phase i be the ith time in the for loop of Algorithm 4. We pessimistically suppose that the noise-oblivious scheme has not found a solution by phase $\log_2(v) - 1$. Then for the next s phases, the proposed variance is at least $2^{\log_2 v} = v$ and the probability that one of these phases is successful is at least $1-\delta^s$. The total number of steps is at most $\sum_{i=0}^{\log_2(v)+s-1} T_\delta(2^i) \leq T_\delta(2^s v)$. ∎

Intuitively, the restriction on T holds whenever T grows at least linearly, which is a reasonable assumption for a run time.

III. Mathematical Preliminaries

In the remainder of this paper, we will study a particular function class (OneMax) and a particular noise distribution (Gaussian, parametrized by the variance).

Let $|x|_1 := |\{i : x_i = 1\}|$. The classical OneMax function is defined as $\mathrm{OM}(x) = |x|_1$, which measures the Hamming distance of its argument to 0^n and is to be maximized. We mention here that our results also hold in the setting of an arbitrary OneMax-target (i.e., the Hamming distance to some predefined string $z \in \{0, 1\}^n$) but adopt the classical definition for clarity of presentation. Let $\sigma^2 \geq 0$. We define the *noisy OneMax function* $\mathrm{OM}_{[\sigma^2]} : \{0, 1\}^n \to \mathbb{R} := x \mapsto |x|_1 + Z$, where Z is a normally distributed random variable $Z \sim \mathcal{N}(0, \sigma^2)$ with zero mean and variance σ^2.

The following proposition gives tail bounds for Z by using estimates of the complementary error function. This will be useful in our proofs for bounding the probability that the $\mathrm{OM}_{[\sigma^2]}$ correctly ranks two arbitrary search points.

Proposition 2: Let Z be a zero-mean Gaussian random variable with variance σ^2. For all $t > 0$ we have

$$\frac{1}{4}\sqrt{\frac{2e}{\pi}} \exp\left(\frac{-t^2}{\sigma^2}\right) \leq \Pr(Z < -t) \leq \frac{1}{2}\exp\left(\frac{-t^2}{2\sigma^2}\right)$$

and, for $x := (t/\sigma\sqrt{2})$

$$\frac{\exp(-x^2)}{\sqrt{\pi\left(x + \sqrt{x^2 + 2}\right)}} < \Pr(Z < -t) \leq \frac{\exp(-x^2)}{\sqrt{\pi\left(x + \sqrt{x^2 + \frac{4}{\pi}}\right)}}.$$

Proof: For a Gaussian distribution, the tail bound of Z is given by $\Pr(Z < -t) = (1/2)\,\mathrm{erfc}((t/\sigma\sqrt{2}))$. Chang *et al.* [9] bounded $\mathrm{erfc}(x)$ by a function $f(x) = \alpha\exp(-\beta x^2)$ for the correct choice of constants α and β. For the upper bound we use $\alpha = \beta = 1$ and the lower bound $\beta = 2$ and $\alpha = (\sqrt{2e}/\pi) \cdot \sqrt{\beta - 1}/\beta$.

For the second bounds, we use the ones given in [32]

$$\frac{2\exp(-x^2)}{\sqrt{\pi\left(x + \sqrt{x^2 + 2}\right)}} < \mathrm{erfc}(x) \leq \frac{2\exp(-x^2)}{\sqrt{\pi\left(x + \sqrt{x^2 + \frac{4}{\pi}}\right)}}.$$

∎

To prove our main result, we will need the following two drift theorems. A drift describes a bias in a randomly moving process. The negative drift theorem gives an upper bound on the probability that a random walk that moves, in expectation, away from the target will nevertheless reach it in at most s steps.

Theorem 1 (Negative Drift [25]): Let $\{X_t : t \in \mathbb{N}_0\}$ be a sequence of random variables over $\mathbb{R}$ with finite expectation. Let $X_0 \leq 0$, $n > 0$, and let T be the random variable that denotes the earliest point in time $t \geq 0$ such that $X_t \geq n$.

Suppose there are $c, 0 < c < n$ and $\varepsilon < 0$ such that, for all t:

1) $\mathrm{E}(X_{t+1} - X_t | T > t, X_t) \leq \varepsilon$;
2) $|X_t - X_{t+1}| < c$.

Then, for all $s \geq 0$

$$\Pr(T \leq s) \leq s \exp\left(-\frac{n|\varepsilon|}{16c^2}\right).$$

The multiplicative drift theorem yields an upper bound if a random walk drifts toward the goal with a speed relative to the position of the process. The tail bounds gives an upper bound on how likely it is that this process has *not* reached its goal within the denoted time.

Theorem 2 (Tail Bounds for Multiplicative Drift [12], [26]): Let $\{X_t : t \in \mathbb{N}_0\}$ be a sequence of random variables over a set $S \subseteq \{0\} \cup [x_{\min}, x_{\max}]$ where $x_{\min} > 0$. Let T be the random variable that denotes the earliest point in time $t \geq 0$ such that $X_t = 0$. If there exists $0 < \delta < 1$ such that $\mathrm{E}(X_t - X_{t+1} | T > t, X_t) \geq \delta X_t$, then

$$\Pr\left(T > \frac{\lambda + \ln(X_0/x_{\min})}{\delta} \,\middle|\, X_0\right) \leq e^{-\lambda} \text{ for all } \lambda > 0.$$

The following lemma is due to von Bahr and Esseen [34] and states an exact equality for the first absolute moment of a random variable Z in terms of its characteristic function $\varphi_Z(t) = \mathrm{E}(e^{itZ})$.

Lemma 1 (Special Case of Lemma 2 of [34]): Let Z be a random variable with $\mathrm{E}(|Z|) < \infty$. Then

$$\mathrm{E}(|Z|) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{1 - \Re\big(\varphi_Z(t)\big)}{t^2} \, \mathrm{d}t$$

where $\Re(z)$ is the real part of $z \in \mathbb{C}$.

We say that a sequence of events $(\mathcal{E}_n)_{n \in \mathbb{N}}$ occurs w.h.p. if the complementary event $\overline{\mathcal{E}}_n$ occurs with a probability of at most $O(1/\operatorname{poly}(n))$ for a polynomial $\operatorname{poly}(n)$.

IV. Formal Analyses

We derive rigorous bounds on the optimization time, defined as the first hitting time of the process to the *true* optimal solution (1^n) of $\mathrm{OM}_{[\sigma^2]}$, on mutation-only approaches and the cGA.

A. Simple Hillclimber

As a warm-up, we begin by showing that the hillclimber introduced in Section II (Algorithm 1) is sensitive to very low noise intensities. This result is not surprising, but serves as a gentle introduction to some of the techniques we will use later in this paper.

Theorem 3: Consider the optimization of $\mathrm{OM}_{[\sigma^2]}$ by RLS. For any constant $\sigma^2 \geq 2$, the true optimum will *not* be evaluated after polynomially many iterations w.h.p.

To prove Theorem 3 we will consider the sequence of values that correspond to the true fitness value of the current solution. In particular, this sequence is a stochastic process over the natural numbers $\{0, 1, \ldots, n\}$. We will prove that for any constant noise intensity, the drift of this process is biased *away* from the optimum. Using the negative drift theorem, we can lower bound the time until the true optimum is hit.

Proof of Theorem 3: Let $\{X_t : t \in \mathbb{N}_0\}$ be the sequence of random variables corresponding to the true OneMax value of the current solution x. This corresponds to the number of ones in the current solution in iteration t.

Let $T = \inf\{t \geq 0 : X_t = n\}$ be the first iteration that RLS generates the true optimum. We begin by showing that, w.h.p., X_0 is far from the optimum. In particular, in line 1 of Algorithm 1, each bit of x is zero (one) with probability $1/2$. Thus, $\mathrm{E}(X_0) = n/2$ and by Chernoff bounds [11], with probability $1 - e^{-\Omega(n)}$, $X_0 \leq 7n/8$. Thus, X_t must be increased by at least $n/8$ before reaching the true optimum. However, we will show that even a constant noise intensity is enough to cause random-walk like behavior so that it is very unlikely that the process gets near the optimum after a polynomial number of steps.

We consider the drift of $\{X_t : t \in \mathbb{N}_0\}$, and we condition the process on $X_t \geq 7n/8$. This has the effect of not counting any steps in the sequence that are too far away from the true optimum. Such an assumption can only underestimate the true value for T.

Let $\mathcal{E}$ be the event that the noisy function misclassifies x and y during the selection step in line 4 of Algorithm 1. In other words, $\mathcal{E}$ is the event that $|x|_1 > |y|_1$ but $\mathrm{OM}_{[\sigma^2]}(x) < \mathrm{OM}_{[\sigma^2]}(y)$ and vice versa.

In iteration $t+1$, the change in X_t is in the right direction if a zero bit is flipped to create y and the noisy function correctly ranks x and y in line 4 of Algorithm 1. The probability of this is $(1 - \Pr(\mathcal{E}))|x|_0/n$. On the other hand, the change in X_t is in the wrong direction if a one bit is flipped to create y, but the noisy function incorrectly ranks y as being superior. The probability of this is $\Pr(\mathcal{E})|x|_1/n$. Putting these two cases together, the drift can be calculated as

$$\begin{aligned}
\mathrm{E}(X_{t+1} - X_t | X_t) &= \frac{|x|_0}{n}(1 - \Pr(\mathcal{E})) - \frac{|x|_1}{n}\Pr(\mathcal{E}) \\
&= \frac{n - X_t}{n} - \Pr(\mathcal{E}) \\
&\leq \frac{n - X_t}{n} - \frac{1}{4}\sqrt{\frac{2e}{\pi}} \exp\left(-1/\sigma^2\right)
\end{aligned}$$

where we have used Proposition 2. Using the fact that $e^z \geq 1 + z$ and that we condition on $X_t \geq 7n/8$, the drift in the right direction is at most

$$\frac{1}{8} - \frac{1}{4}\sqrt{\frac{2e}{\pi}}\left(1 - \frac{1}{\sigma^2}\right) \leq -0.0394.$$

Here, we have used the fact that $\sigma^2 \geq 2$ and so $(1 - \sigma^{-2}) \geq 1/2$. Therefore, $\mathrm{E}(X_{t+1} - X_t | X_t) \leq \varepsilon$ for a constant $\varepsilon < 0$. Applying Theorem 1, we conclude that w.h.p., T is superpolynomial in n. ∎

Thus in the case of the hillclimber without any noise-handling mechanism, even a small constant noise level obscures any useful selection information, and the algorithm resorts to random walk behavior. We point out here that a similar result holds for the $(1 + 1)$ EA [18].

B. Population-Based EA

A common approach to adapt an EA to a noisy environment is to increase the population size [3], [20]. The idea is that

using more individuals allow for some kind of implicit resampling of points during optimization. However, we will prove in this section that for mutation-only EAs, increasing the population cannot help as the noise intensity grows large. Specifically, there is a threshold point in the noise intensity above which no polynomial size population can optimize $\textsc{om}_{[\sigma^2]}$ efficiently.

We consider the $(\mu+1)$ EA as introduced in Algorithm 2. We will first, in Theorem 4, give a sufficient condition for when a noise model is intractable for optimization by the $(\mu+1)$ EA. While uniform selection removes any individual from the population with probability $1/(\mu+1)$, the condition of Theorem 4 requires that the noise is strong enough so that the $(\mu+1)$ EA will remove any individual with at least half that probability. Then we will show that, in the case of additive noise sampled from a Gaussian distribution, this condition is fulfilled if the noise is large enough, showing that the $(\mu+1)$ EA cannot deal with arbitrary Gaussian noise (see Corollary 1).

Theorem 4: Let $\mu \geq 1$ and D a random variable on $\mathbb{R}$ (which will be used as noise). Let Y be the random variable describing the minimum over μ independent copies of D. Suppose

$$\Pr(Y > D + n) \geq \frac{1}{2(\mu+1)}.$$

Consider optimization of OneMax with additive noise from D by $(\mu+1)$ EA. Then, for μ bounded from above by a polynomial, the optimum will *not* be evaluated after polynomially many iterations w.h.p.

Proof: This proof will show by induction that the expected number of the *proportion of individuals* with many 1 is small—exponentially small in the number of 1s. The induction step considers all possibilities of creating individuals with a given number of 1s and the probability of removing an individual with that number of 1s from the population.

To this end, for all t and all $i \leq n$, let X_i^t be the random variable describing the proportion of individuals in the population of iteration t with exactly i 1s. Let $c = 800$, $b = 20$, $a = (c-1)/c$, and $a' = (c-2)/c$. We show by induction on t that

$$\forall t, i \geq an : \ \mathrm{E}\big(X_i^t\big) \leq b^{an-i}.$$

In other words, the expected number of individuals with i 1s is decaying exponentially with i after an. This will give the desired result with a simple union bound over polynomially many time steps.

The claim holds at the start of the algorithm with an application of Hoeffding's inequality for the number of 1s in a random individual. Fix some value t and suppose the claim holds for that t. Let some value $i \geq an$ be given and let $x = b^{an-i}$. We will now show $\mathrm{E}(X_i^{t+1}) \leq x$ by considering one generation of the $(\mu+1)$ EA.

We distinguish four cases depending on whether an individual with less than $a'n$ 1s has been selected for reproduction, with $i-k$ 1s for some k with $1 \leq k \leq n/c$, with exactly i 1s or with strictly more than i 1s. For each of these cases we estimate the number of individuals that can be chosen to reproduce, as well as the probability for such an individual to produce an offspring with exactly i 1s. The following table gives upper bounds for both values in all four cases; we will justify all these values below.

	Proportion	**Probability**
$< a'n$	1	$2^{-\Omega(n \ln n)}$
$= i-k$	xb^k	$(2/c)^k$
$= i$	x	$1/\mathrm{e} + 1/(c-1)$
$> i$	$x/(b-1)$	1

Clearly the proportion of individuals with $< a'n$ 1s is bounded from above by 1; for such an individual with m 0s, at least half of these 0s need to flip, which has a probability of at most $2^m/n^{m/2} = 2^{-\Omega(n \ln n)}$, using $m \geq n/c$. For any $k < n/c$, we get a bound of xb^k for the number of individuals with exactly $i-k$ 1s from the induction hypothesis; as these individuals have at most $2n/c$ many 0s, the probability of flipping at least k of these to 1 is $\leq (2/c)^k$. For an individual with exactly i 1s to create an offspring with exactly i 1s, we can either not flip any bit (with a probability tending to $1/e$) or we flip as many 1s as 0s; flipping k 1s has a probability of at most $1/c$ (as $i \geq an$), thus we can bound the probability of creating an offspring with exactly i 1s by

$$1/e + \sum_{k=1}^{\infty} c^{-k} = 1/e + 1/c - 1.$$

With a similar geometric sum we get that the number of individuals with $> i$ 1s is, using the induction hypothesis, at most $x/(b-1)$.

From the table we can now deduce that the probability of producing an offspring with exactly i 1s in iteration t is at most

$$2^{-\Omega(n \ln n)} + x\left(\frac{1}{e} + \frac{1}{c-1} + \frac{1}{b-1} + \sum_{k=1}^{\infty}\left(\frac{2b}{c}\right)^k\right).$$

Using $x \geq b^{-n/c}$ we see that $2^{-\Omega(n \ln n)}$ has asymptotically no impact on the sum. Furthermore, from our choice of b and c, we have

$$\frac{1}{e} + \frac{1}{c-1} + \frac{1}{b-1} + \frac{1}{\frac{c}{2b}-1} < \frac{1}{2}.$$

Thus, we have that we get less than $x/2$ individuals with exactly i 1s in expectation, while the premise of the theorem gives that any individual has a probability of at least $1/2$ to die in any given iteration. This shows that $\mathrm{E}(X_i^{t+1})$ cannot go above x. ∎

We apply Theorem 4 to show that large noise levels make it impossible for the $(\mu+1)$ EA to efficiently optimize when the noise is significantly larger than the range of objective values. The proof is a simple exercise in bounding the tails of a Gaussian distribution using Proposition 2.

Corollary 1: Consider optimization of $\textsc{om}_{[\sigma^2]}$ by the $(\mu+1)$ EA with $\mu = \omega(1)$ and μ bounded from above by a polynomial in n. Suppose $\sigma^2 \geq (na)^2$, for some $a = \omega(1)$. Then the optimum will *not* be evaluated after polynomially many iterations w.h.p. In particular, the $(\mu+1)$ EA does not scale gracefully with noise.

Proof: We set up to use Theorem 4. Let $D \sim \mathcal{N}(0, \sigma^2)$ and let Y be the minimum over μ independent copies of D. We want to bound $\Pr(D+n < Y)$. We lower bound this probability by a case distinction into two events. To that end, we choose two points $t_0 < t_1 \leq -3\sigma < 0$ such that $\Pr(D < t_0) = 0.8/\mu$ and $\Pr(D < t_1) = 2/\mu$. Note that $-t_0/\sigma > -t_1/\sigma \geq 3$. Hence, the maximum value for the second lower bound on $\Pr(D < t)$ as in Proposition 2, for $t \leq -3\sigma$, is a constant, whereas $\Pr(D < t_0) < \Pr(D < t_1) = o(1)$, since $\mu = \omega(1)$. Thus, there actually exist $t_0, t_1 < 0$ as wanted. We define the following two events.

1) The event that $D < t_0 - n$ and $t_0 < Y$.
2) The event that $t_0 - n < D < t_1 - n$ and $t_1 < Y$.

Clearly, the events A and B are disjoint and are contained in the event that $D + n < Y$. Both events are made up of two independent events, the first one making a statement about D, the second one making a statement about Y. Let the former be indexed by D and the latter by Y. Thus, $A = A_D \cap A_Y$ and $B = B_D \cap B_Y$.

The probabilities of the events indexed with Y can be easily calculated by the μ independent events that none of the μ copies of D was smaller than t_0, since then the minimum would trivially also not be smaller. Hence, we get:

1) $\Pr(A_Y) = (1 - \Pr(Y < t_0))^\mu = (1 - 0.8/\mu)^\mu$;
2) $\Pr(B_Y) = (1 - \Pr(Y < t_1))^\mu = (1 - 2/\mu)^\mu$.

We now focus on the events indexed by D. By our choice of t_0, we have $t_0 < -3\sigma \leq -3na < -na$. We then transform

$$t_0 \leq -na \iff \frac{t_0}{a} \leq -n \iff t_0 + \frac{t_0}{a} \leq t_0 - n$$

hence, $t_0(1 + 1/a) \leq t_0 - n$. We can now bound $\Pr(A_D) = \Pr(D < t_0 - n)$ asymptotically from both sides. The upper bound is trivial

$$\Pr(D < t_0 - n) \leq \Pr(D < t_0) = \frac{0.8}{\mu}.$$

The lower bound is as follows, making use of $t_0 - n \geq t_0(1 + 1/a)$:

$$\begin{aligned}
\Pr(D < t_0 - n) &\geq \Pr\left(D < t_0\left(1 + \frac{1}{a}\right)\right) \\
&> \frac{\exp\left(-\frac{t_0^2\left(1+\frac{1}{a}\right)^2}{2\sigma^2}\right)}{\sqrt{\pi\left(-\frac{t_0\left(1+\frac{1}{a}\right)}{\sigma\sqrt{2}} + \sqrt{\frac{t_0^2\left(1+\frac{1}{a}\right)^2}{2\sigma^2} + 2}\right)}} \\
&> \frac{\exp\left(-\frac{t_0^2}{2\sigma^2}\right)\exp\left(\left(1+\frac{1}{a}\right)^2\right)}{\sqrt{\pi\left(1+\frac{1}{a}\right)\left(-\frac{t_0}{\sigma\sqrt{2}} + \sqrt{\frac{t_0^2}{2\sigma^2} + 2}\right)}} \\
&> \frac{e^{-o(1)}}{1+o(1)} \cdot \frac{\exp\left(-\frac{t_0^2}{2\sigma^2}\right)}{\sqrt{\pi\left(-\frac{t_0}{\sigma\sqrt{2}} + \sqrt{\frac{t_0^2}{2\sigma^2} + 2}\right)}} \\
&> \frac{e^{-o(1)}}{1+o(1)} \cdot \frac{\exp\left(-\frac{t_0^2}{2\sigma^2}\right)}{\sqrt{\pi\left(-\frac{t_0}{\sigma\sqrt{2}} + \sqrt{\frac{t_0^2}{2\sigma^2} + \frac{4}{\pi}}\right)}} \\
&> (1 - o(1)) \cdot \frac{0.984 \cdot \exp\left(-\frac{t_0^2}{2\sigma^2}\right)}{\sqrt{\pi\left(-\frac{t_0}{\sigma\sqrt{2}} + \sqrt{\frac{t_0^2}{2\sigma^2} + \frac{4}{\pi}}\right)}} \\
&> 0.98 \cdot \Pr(D < t_0)
\end{aligned}$$

because

$$\frac{1}{\sqrt{\pi\left(\frac{x}{\sqrt{2}} + \sqrt{\frac{x^2}{2} + 2}\right)}} > \frac{0.984}{\sqrt{\pi\left(\frac{x}{\sqrt{2}} + \sqrt{\frac{x^2}{2} + \frac{4}{\pi}}\right)}}$$

for $x \geq 3$, and because $1/a = o(1)$, as we assume $a = \omega(1)$. Thus, we get $\Pr(D < t_0 - n) = \Pr(A_D) \leq 1/\mu$ and $\Pr(A_D) > 0.98 \cdot 0.8/\mu = 0.784/\mu$.

We proceed in a similar way for bounding $\Pr(B_D) = \Pr(t_0 - n < D < t_1 - n) = \Pr(D < t_1 - n) - \Pr(D < t_0 - n)$.

Since we already have bounds for $\Pr(D < t_0 - n)$, we only have to bound $\Pr(D < t_1 - n)$. These calculations are analogous to the ones beforehand and do not change anything asymptotically, thus, we get $\Pr(D < t_1 - n) \geq \Pr\left(D < t_1(1 + 1/a)\right) > 0.98 \cdot 2/\mu$, and therefore in total $\Pr(t_0 - n < D < t_1 - n) = \Pr(B_D) > 0.98 \cdot 2/\mu - 0.8/\mu = 1.16/\mu$.

Overall, we get

$$\begin{aligned}
\Pr(Y > D + n) &\geq \Pr(A) + \Pr(B) \\
&= \frac{0.784}{\mu}\left(1 - \frac{0.8}{\mu}\right)^\mu + \frac{1.16}{\mu}\left(1 - \frac{2}{\mu}\right)^\mu \\
&\geq \frac{0.784}{\mu}e^{-0.8} + \frac{1.16}{\mu}e^{-2} \geq \frac{1}{2\mu} \geq \frac{1}{2(\mu+1)}
\end{aligned}$$

where we made use of $(1 - x)^t \geq (1 - o(1))e^{-xt}$ if $x^2 t = o(1)$ [29].

Finally, we can use Theorem 4, since μ is bounded from above by a polynomial. This completes the proof. ■

Note that the condition $\sigma^2 = \omega(n^2)$ comes from the very pessimistic assumption of one good individual having to lose against many much worse individuals; thus we conjecture that already much smaller noise levels will lead to inefficient optimization.

C. Compact GA

We now show a positive result for the cGA EDA; specifically, that it can scale gracefully with Gaussian noise in the sense of Definition 1. Let $T^\star$ be the optimization time of the cGA on $\textsc{om}_{[\sigma^2]}$, namely, the first time that it generates the underlying true optimal solution 1^n. We consider the stochastic process $\{X_t : t \in \mathbb{N}_0\}$ defined as follows:

$$X_t = n - \sum_{i=1}^{n} p_{i,t} \tag{1}$$

and bound the optimization time by $T = \inf\{t \in \mathbb{N}_0 : X_t = 0\}$. This is the time until the product distribution has converged to

the optimal frequency vector. Clearly $T^\star \leq T$ since the cGA produces 1^n in the Tth iteration almost surely. However, $T^\star$ and T can be infinite when there is a $t < T^\star$, where $p_{i,t} = 0$ since the process can never subsequently generate any string x with $x_i = 1$. To circumvent this, Droste [14] estimates $\mathrm{E}(T^\star)$ conditioned on the event that $T^\star$ is finite, and then bounds the probability of finite $T^\star$. In this paper, we will prove that as long as K is large enough, the optimization time is finite (indeed, polynomial) w.h.p.

We first define the probability that the cGA misclassifies two points under comparison due to the presence of noise.

Definition 2: Let $x, y \in \{0,1\}^n$. Without loss of generality, suppose $|x|_1 - |y|_1 = \ell \geq 0$. Since $\mathrm{OM}_{[\sigma^2]}$ is a function of unitation, the probability Δ that it misclassifies y as superior to x depends only on the so-called phenotypic distance ℓ. We define $\Delta : [n] \cup \{0\} \to [0,1]$ as

$$\Delta(\ell) = \begin{cases} 1/2 & \ell = 0 \\ \Pr\big(\mathcal{E} \big| |x|_1 - |y|_1 = \ell\big) & \ell > 0 \end{cases}$$

where $\mathcal{E}$ is the event that $\mathrm{OM}_{[\sigma^2]}(x) < \mathrm{OM}_{[\sigma^2]}(y)$ and $[n] \coloneqq [1,n] \cap \mathbb{N}$.

The following lemma bounds the misclassification probability in terms of the noise intensity measured by the variance.

Lemma 2: For any $\ell \in [n]$, $\Delta(\ell) > \Delta(\ell+1)$. Moreover, assuming $\sigma^2 > 0$

$$\Delta(\ell) \leq \frac{1}{2}\Big(1 - \Theta\Big(\sigma^{-2}\Big)\Big).$$

Proof: Let x and y be chosen arbitrarily from the set of all length-n binary strings pairs with $|x|_1 - |y|_1 = \ell$ for any $\ell \in [n]$. The event that $\mathrm{OM}_{[\sigma^2]}$ incorrectly classifies y as superior to x is equivalent to the event $\mathrm{OM}_{[\sigma^2]}(x) < \mathrm{OM}_{[\sigma^2]}(y)$

$$\Pr\big(\mathrm{OM}_{[\sigma^2]}(x) < \mathrm{OM}_{[\sigma^2]}(y)\big) = \Pr(\ell + (Z_1 - Z_2) < 0)$$

where $Z_1, Z_2 \sim \mathcal{N}(0, \sigma^2)$ are independent identically distributed. Letting $Z^* \coloneqq Z_1 - Z_2$, we have $Z^* \sim \mathcal{N}(0, 2\sigma^2)$ and $\Delta(\ell) = \Pr(Z^* < -\ell)$. Furthermore, $\Delta(\ell+1) = \Pr(Z^* < -(\ell+1)) < \Pr(Z^* < -\ell) = \Delta(\ell)$. Finally, applying Proposition 2, we have $\Pr(Z^* < -\ell) \leq (1/2)e^{-\ell^2/(4\sigma^2)} \leq (1/2)e^{-1/(4\sigma^2)}$. The proof is then completed by applying the well-known bound $1 - z > e^{-z/(1-z)}$ [28] and setting $z = 1/(4\sigma^2+1)$. ■

We need the following technical lemma that yields some properties of the sum of n independent random variables over $\{-1, 0, 1\}$. Specifically, we derive a bound on the probability that the sum balances (i.e., is equal to zero) and a lower bound on the first absolute moment of the sum. We will later use these results in proofs about the drift of the allele frequencies during the run of the cGA.

Lemma 3: Let $0 < a < 1$ be a constant. Consider a random variable $Z = Z_1 + Z_2 + \cdots + Z_n$, each Z_i independent

$$Z_i = \begin{cases} 1 & \text{with probability } p_i(1-p_i) \\ -1 & \text{with probability } p_i(1-p_i) \\ 0 & \text{with probability } 1 - 2p_i(1-p_i) \end{cases}$$

with $a \leq p_i \leq 1$ for every $i \in \{1, \ldots, n\}$. Then $\Pr(Z = 0) \geq 1/(4\sqrt{n})$, and

$$\mathrm{E}(|Z|) \geq a\sqrt{2/n}\left(n - \sum_{i=1}^{n} p_i\right).$$

Proof: Let $\xi = |Z_1| + |Z_2| + \cdots + |Z_n|$. Then ξ is distributed as a Poisson binomial distribution with each success probability equal to $2p_i(1-p_i)$. Furthermore, $Z = 0$ when exactly k of the Z_i variables are nonzero for some even k, and exactly $k/2$ of these are selected to be negative with the remaining $k/2$ positive. The probability that exactly k of the variables are nonzero is $\Pr(\xi = k)$, and the probability of selecting exactly $k/2$ to be positive is $\binom{k}{k/2}2^{-k}$ so we can write

$$\Pr(Z = 0) = \sum_{k=0}^{n} \Pr(\xi = k)\binom{k}{k/2}2^{-k}$$

where $\binom{k}{k/2} = 0$ if $k \equiv 1 \pmod 2$. Since $\binom{k}{k/2}$ vanishes at odd i, we have

$$\Pr(Z = 0) = \sum_{k=0}^{\lfloor n/2 \rfloor} \Pr(\xi = 2k)\binom{2k}{k}2^{-2k}.$$

$\binom{2k}{k}$ is the kth central binomial coefficient, for which we have the well-known bound $(2^{2k}/\sqrt{4k}) \leq \binom{2k}{k}$ (see [24]), hence

$$\begin{aligned} \Pr(Z = 0) &\geq \Pr(\xi = 0) + \sum_{k=1}^{\lfloor n/2 \rfloor} \Pr(\xi = 2k)\frac{1}{2\sqrt{k}} \\ &\geq \frac{1}{2\sqrt{n}} \Pr(\xi \text{ is even}) \end{aligned} \tag{2}$$

since $(1/(2\sqrt{n})) \leq (1/(2\sqrt{k})) \leq 1$. To finish the proof, note that for any integer random variable X, $\Pr(X \text{ is even}) = (1 + G(-1))/2$, where $G(z) = \mathrm{E}(z^X)$ is the probability generating function for X. Since ξ is a Poisson binomial distribution with success probabilities $q_1, q_2, \ldots, q_n$, we can write the probability generating function as $G(z) = \prod_{i=1}^{n}(1 - q_i + q_i z)$. Therefore

$$\begin{aligned} \Pr(\xi \text{ is even}) &= (1 + G(-1))/2 \\ &= \frac{1}{2}\left(1 + \prod_{i=1}^{n}(1 - 2q_i)\right). \end{aligned}$$

Finally, since $q_i = 2p_i(1-p_i) \leq 1/2$ for all $i \in \{1, \ldots, n\}$, the product in the above equation must be non-negative. Therefore, $\Pr(\xi \text{ is even}) \geq 1/2$ and the claimed bound on $\Pr(Z = 0)$ follows from (2).

We now bound the first absolute moment of Z from below. For every $S \subseteq [n]$, denote as $\mathcal{E}_S$ the event that $|Z_i| = 1 \iff i \in S$. We first calculate the expectation of $|Z|$ conditioned on these events. Since the probabilities p_i are mutually independent

$$\mathrm{E}\big(e^{itZ} \big| \mathcal{E}_S\big) = e^{it\mathrm{E}(Z|\mathcal{E}_S)} = \prod_{j=1}^{n} e^{it\mathrm{E}\left(Z_j \middle| \mathcal{E}_S\right)}.$$

If $j \in S$, then $Z_j = 1$ with probability $1/2$ and $Z_j = -1$ with probability $1/2$. Thus, $e^{it\mathrm{E}(Z_j|\mathcal{E}_S \wedge j \in S)} = e^{it}/2 + e^{-it}/2$. On the

other hand, if $j \notin S$, then $Z_j = 0$ and so $e^{itE(Z_j|\mathcal{E}_S \wedge j \notin S)} = e^0 = 1$. Denote as $\mathbf{1}_S$ the indicator function for S. We collect the above terms to write

$$\begin{aligned}\mathrm{E}\big(e^{itZ}\big|\mathcal{E}_S\big) &= \prod_{j=1}^{n}\left(\mathbf{1}_S(j)\left(\frac{e^{it}}{2}+\frac{e^{-it}}{2}\right)+1-\mathbf{1}_S(j)\right)\\ &= \prod_{j\in S}\cos t = (\cos t)^{|S|}.\end{aligned}$$

So by Lemma 1

$$\mathrm{E}(|Z||\mathcal{E}_S) = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{1-(\cos t)^{|S|}}{t^2}\,\mathrm{d}t = g(|S|)$$

where $g(k) = 2\lceil k/2\rceil\binom{2\lceil k/2\rceil}{\lceil k/2\rceil}2^{-2\lceil k/2\rceil}$ (see Lemma 6 in the Appendix for a detailed derivation). Again applying bounds on the central binomial coefficient, $g(k) \geq \sqrt{\lceil k/2\rceil} \geq \sqrt{k/2}$. By the law of total expectation

$$\begin{aligned}\mathrm{E}(|Z|) &= \sum_{k=1}^{n} g(k)\sum_{S\subseteq[n]:|S|=k}\Pr(\mathcal{E}_S)\\ &\geq \frac{1}{\sqrt{2n}}\sum_{k=1}^{n}k\sum_{S\subseteq[n]:|S|=k}\Pr(\mathcal{E}_S)\\ &= \frac{\mathrm{E}(\xi)}{\sqrt{2n}}.\end{aligned} \tag{3}$$

Since ξ follows a Poisson binomial distribution with the ith success probability equal to $2p_i(1-p_i)$, and every $p_i \geq a$

$$\mathrm{E}(\xi) = \sum_{i=1}^{n}2p_i(1-p_i) \geq 2a\left(n-\sum_{i=1}^{n}p_i\right).$$

Substituting this inequality into (3) completes the proof. ∎

We will use Lemma 3 in the proof of the next lemma whenever we need bounds on the expected absolute difference of the count of ones between the offspring generated in lines 4 and 6 of Algorithm 3. Specifically, the following lemma bounds the drift on X_t as defined in (1), conditioned on the event that no allele frequency gets too small.

Lemma 4: Consider the cGA optimizing $\mathrm{OM}_{[\sigma^2]}$ and let X_t be the stochastic process defined in (1). Assume that there exists a constant $a > 0$ such that $p_{i,t} \geq a$ for all $i \in [n]$ and that $X_t > 0$, then $\mathrm{E}(X_t - X_{t+1}|X_t) \geq \delta X_t$ where $1/\delta = \mathcal{O}(\sigma^2 K\sqrt{n})$.

Proof: Let x and y be the offspring generated in iteration t and $Z_t = |x|_1 - |y|_1$. Then $Z_t = Z_{1,t} + \cdots + Z_{n,t}$ where

$$Z_{i,t} = \begin{cases}-1 & \text{if } x_i = 0 \text{ and } y_i = 1\\ 0 & \text{if } x_i = y_i\\ 1 & \text{if } x_i = 1 \text{ and } y_i = 0.\end{cases}$$

Let $\mathcal{E}$ denote the event that in line 8, the evaluation of $\mathrm{OM}_{[\sigma^2]}$ incorrectly ranks x and y. Without loss of generality, suppose $|x|_1 \geq |y|_1$. Then $\mathrm{E}(X_t - X_{t+1}|X_t, \overline{\mathcal{E}}) = \mathrm{E}(|Z_t|)/K$. On the other hand, if $\mathrm{OM}_{[\sigma^2]}(x)$ evaluates to at most $\mathrm{OM}_{[\sigma^2]}(y)$ during iteration t, the roles above are swapped and $\mathrm{E}(X_t - X_{t+1}|X_t, \mathcal{E}) = -\mathrm{E}(|Z_t|)/K$. By the law of total expectation

$$\mathrm{E}(X_t - X_{t+1}|X_t) = \frac{\mathrm{E}(|Z_t|)}{K}(1 - 2\Pr(\mathcal{E})). \tag{4}$$

For any $i \in [n]$, $\Pr(Z_{i,t} = 1) = \Pr(Z_{i,t} = -1) = p_{i,t}(1-p_{i,t})$ and $\Pr(Z_{i,t} = 0)$ is the inverse. Since we have assumed each $p_{i,t} \geq a$, we can apply Lemma 3 to obtain

$$\mathrm{E}(|Z_t|) \geq a\sqrt{\frac{2}{n}}\left(n - \sum_{i=1}^{n}p_{i,t}\right) = aX_t\sqrt{2/n}. \tag{5}$$

To complete the proof, we substitute the inequality in (5) into (4) and use Lemma 2 to bound $\Pr(\mathcal{E}) = \Delta(||x|_1 - |y|_1|)$ from above. ∎

To use Lemma 4, we require that the allele frequencies stay large enough during the run of the algorithm. Increasing the effective population size K obviously translates to finer-grained allele frequency values, which means slower dynamics for $p_{i,t}$. We point out that this scaling of the update probabilities is an application of the technique of *rescaled mutations* [3] to the space of probability vectors. Provided that K is set sufficiently large, the allele frequencies remain above an arbitrary constant for any polynomial number of iterations with very high probability. This is captured by the following lemma.

Lemma 5: Consider the cGA optimizing $\mathrm{OM}_{[\sigma^2]}$ with $\sigma^2 > 0$. Let $0 < a < 1/2$ be an arbitrary constant and $T' = \min\{t \geq 0 : \exists i \in [n], p_{i,t} \leq a\}$. If $K = \omega(\sigma^2\sqrt{n}\log n)$, then for every polynomial $\mathrm{poly}(n)$, n sufficiently large, $\Pr(T' < \mathrm{poly}(n))$ is superpolynomially small.

Proof: Let $i \in [n]$ be arbitrary. Let $\{Y_t : t \in \mathbb{N}_0\}$ be the stochastic process $Y_t = (1/2 - p_{i,t})K$. We first argue that

$$\mathrm{E}(Y_t|Y_1,\ldots,Y_{t-1}) \leq Y_{t-1} - \Omega\left(\sigma^{-2}\right)\frac{\Pr(x_i \neq y_i)}{\sqrt{n}}. \tag{6}$$

Let x and y be the strings generated in iteration t of the cGA (lines 4 and 6 of Algorithm 3). We define $\hat{x} := (x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ to be the substring of x constructed by removing the ith element and $\hat{y}$ similarly. Since each element of x and y is constructed independently, we can regard $\hat{x}$, $\hat{y}$, x_i, and y_i to be independent.

Define the random variable $\delta_t := Y_t - Y_{t-1}$. Note that $\mathrm{E}(Y_t|Y_1,\ldots,Y_{t-1}) = Y_{t-1} + \mathrm{E}(\delta_t)$ where $\delta_t \in \{-1, 0, 1\}$. Define $\hat{\ell} = |\hat{x}|_1 - |\hat{y}|_1$. We distinguish between the two events that $|\hat{\ell}|$ is nonzero or zero.

Case $|\hat{\ell}| > 0$: Suppose without loss of generality that $\hat{\ell} > 0$ (i.e., $|\hat{x}|_1 > |\hat{y}|_1$). So, $\delta_t = 0$ if and only if $x_i = y_i$. Moreover, $\delta_t = -1$ only in the event that: 1) $x_i = 1$ and $y_i = 0$ and x is accepted (in which case $\ell = \hat{\ell} + 1$) or 2) $x_i = 0$ and $y_i = 1$ and x is *not* accepted (in which case $\ell = \hat{\ell} - 1$). Event 1) occurs only if $\mathrm{OM}_{[\sigma^2]}$ does not misclassify x and y, whereas event 2) occurs only if $\mathrm{OM}_{[\sigma^2]}$ does misclassify x and y. Thus, $\Pr(\delta_t = -1) = \Pr(x_i = 1, y_i = 0)(1 - \Delta(\hat{\ell} + 1)) + \Pr(x_i = 0, y_i = 1)\Delta(\hat{\ell} - 1)$.

Similarly, $\delta_t = 1$ only in the event that: 1) $x_i = 1$ and $y_i = 0$ but x is *not* accepted, because x and y were misclassified by $\mathrm{OM}_{[\sigma^2]}$ or 2) $x_i = 0$ and $y_i = 1$ and x is accepted because $\mathrm{OM}_{[\sigma^2]}$ ranked x and y correctly. Thus, $\Pr(\delta_t = 1) = \Pr(x_i = 1, y_i = 0)\Delta(\hat{\ell} + 1) + \Pr(x_i = 0, y_i = 1)(1 - \Delta(\hat{\ell} - 1))$. Since $\Pr(x_i = 1, y_i = 0) = \Pr(x_i = 0, y_i = 1) = \Pr(x_i \neq y_i)/2$

$$\begin{aligned}\mathrm{E}(\delta_t) &= \Pr(\delta_t = 1) - \Pr(\delta_t = -1)\\ &= -\Pr(x_i \neq y_i)\left(\Delta\left(\hat{\ell} - 1\right) - \Delta\left(\hat{\ell} + 1\right)\right) < 0\end{aligned}$$

where we apply Lemma 2. We conclude that in this case

$$\mathrm{E}\Big(Y_t|\hat{\ell} \neq 0, Y_1, \ldots, Y_{t-1}\Big) = Y_{t-1} + \mathrm{E}(\delta_t) < Y_{t-1}.$$

Case $\hat{\ell} = 0$: In this case, if $x_i = y_i$, then $x = y$ and there is zero drift. Otherwise, $x_i > y_i$ and so $|x|_1 - |y|_1 = 1$, or $y_i > x_i$ and $|y|_1 - |x|_1 = 1$. The drift in this case only depends on whether or not $\mathrm{OM}_{[\sigma^2]}$ misclassifies x and y. In particular, $\Pr(\delta_t = -1) = \Pr(x_i = 1, y_i = 0)(1 - \Delta(1)) + \Pr(x_i = 0, y_i = 1)(1 - \Delta(1))$, and $\Pr(\delta_t = 1) = \Pr(x_i = 1, y_i = 0)\Delta(1) + \Pr(x_1 = 0, y_i = 1)\Delta(1)$. By Lemma 2

$$\begin{aligned}\mathrm{E}(\delta_t) &= \Pr(\delta_t = 1) - \Pr(\delta_t = -1)\\ &= -\Pr(x_i \neq y_i)(1 - 2\Delta(1)) \leq -\Omega\Big(\sigma^{-2}\Big)\Pr(x_i \neq y_i).\end{aligned}$$

For this case, $\mathrm{E}(Y_t|\hat{\ell} = 0, Y_1, \ldots, Y_{t-1}) = Y_{t-1} + \mathrm{E}(\delta_t) \leq Y_{t-1} - \Omega(\sigma^{-2})\Pr(x_i \neq y_i)$.

Applying the law of total expectation, $\mathrm{E}(Y_t|Y_1, \ldots, Y_{t-1})$ is bounded above by

$$Y_t - \Omega\Big(\sigma^{-2}\Big)\Pr(x_i \neq y_i)\Pr\Big(\hat{\ell} = 0\Big).$$

It remains to bound $\Pr(\hat{\ell} = 0) = \Pr(|\hat{x}|_1 = |\hat{y}|_1)$. We define a random variable $Z = Z_2 + \cdots + Z_n$ where

$$Z_j = \begin{cases} +1 & \text{if } x_j > y_j \\ 0 & \text{if } x_j = y_j \\ -1 & \text{if } x_j < y_j. \end{cases}$$

So $\Pr(|\hat{x}|_1 = |\hat{y}|_1) = \Pr(Z = 0) \geq 1/(4\sqrt{n-1})$ by Lemma 3 since $0 \leq \Pr(x_j > y_j) = \Pr(x_j < y_j) = p_j(1 - p_j) \leq 1/2$ for all $j \in \{2, \ldots, n\}$, proving the claim in (6).

Note that $\{Y_t : t \in \mathbb{N}\}$ is a Markov chain on the set $\{-K/2, -K/2 + 1, \ldots, K/2 - 1, K/2\}$ with $Y_1 = 0$. Let $T = \min\{t : Y_t > (1/2 - a)K\}$. In any iteration, if $x_i = y_i$, then $Y_t = Y_{t-1}$. Thus, for an estimate of the upper bounds of T, we can ignore self-loops in the chain.

More formally, let $\{\hat{Y}_t : t \in \mathbb{N}\}$ be the restriction of Y_t to iterations such that $Y_t \neq Y_{t-1}$. Similarly, let $\hat{T} = \min\{t : \hat{Y}_t > (1/2 - a)K\}$. The random variable T stochastically dominates the random variable $\hat{T}$ since removing equal moves can only make the process hit faster, i.e., $\forall t \in \mathbb{N} : \Pr(T > t) \geq \Pr(\hat{T} > t)$. Due to the above arguments

$$\begin{aligned}&\mathrm{E}\Big(\hat{Y}_t|\hat{Y}_1, \ldots, \hat{Y}_{t-1}\Big)\\ &\quad= \mathrm{E}\Big(\hat{Y}_t|x_i \neq y_i, \hat{Y}_1, \ldots, \hat{Y}_{t-1}\Big)\\ &\quad= \hat{Y}_t - \mathrm{E}(\delta_t|x_i \neq y_i) \leq \hat{Y}_t - \Omega\Big(\sigma^{-2}/\sqrt{n}\Big).\end{aligned}$$

By Theorem 1, since $Y_1 = \hat{Y}_1 = 0$ and $|\hat{Y}_t - \hat{Y}_{t+1}| = 1 < \sqrt{2}$, for all $s \geq 0$

$$\Pr(T \leq s) \leq \Pr\Big(\hat{T} \leq s\Big) \leq s\exp\left(-\frac{(1/2 - a)K|\varepsilon|}{32}\right)$$

with $\varepsilon = -\Omega(\sigma^{-2}/\sqrt{n})$. Since $K = \omega(\sigma^2\sqrt{n}\log n)$, $\Pr(T \leq s) = sn^{-\omega(1)}$.

So, for any polynomial $s = \mathrm{poly}(n)$, with probability super-polynomially close to one, Y_s has not yet reached a state larger than $(1/2 - a)K$, and so $p_{i,t} > a$ for all $0 \leq t \leq s$. As this holds for arbitrary i, applying a union bound retains a super-polynomially small probability that any of the n frequencies have gone below a by $s = \mathrm{poly}(n)$ steps. ■

It is now straightforward to prove that the optimization time of the cGA is polynomial in the problem size and the noise variance. This is in contrast to the mutation-based $(\mu + 1)$ EA, which fails when the variance becomes large. This means the cGA scales gracefully with noise in the sense of Definition 1 applied to the $\mathrm{OM}_{[\sigma^2]}$ noise model. We actually prove a stronger condition than the first hitting time of the true optimal solution. We show that after polynomial time, the allele frequencies have converged to the optimal frequency distribution ($p_{i,t} = 1$ for all $i \in [n]$). The proof is carried out by first conditioning on the event that no allele frequency gets too small. This event is guaranteed w.h.p. by Lemma 5. The result then follows by showing that the conditional drift of the total allele frequencies in the right direction is large enough to hit the optimal distribution within the claimed time bound.

Theorem 5: Consider the cGA optimizing $\mathrm{OM}_{[\sigma^2]}$ with variance $\sigma^2 > 0$. If $K = \omega(\sigma^2\sqrt{n}\log n)$, then w.h.p., the cGA has converged to the optimal frequency distribution after $\mathcal{O}(K\sigma^2\sqrt{n}\log Kn)$ steps.

Proof: We will consider the drift of the stochastic process $\{X_t : t \in \mathbb{N}_0\}$ corresponding to the total allele frequencies defined as in (1). This is a stochastic process over the state space $S = \{n - i/K\}$ for each $i = 0, \ldots, nK$. Hence, the bounds needed for Theorem 2 are $[x_{\min}, x_{\max}] = [1/K, n]$.

Fix a constant $0 < a < 1/2$. We say the process has *failed* by time t if there exists some $s \leq t$ and some $i \in [n]$ such that $p_{i,s} \leq a$. Let $T = \min\{t \in \mathbb{N}_0 : X_t = 0\}$. Assuming the process never fails, by Lemma 4, the drift of $\{X_t : t \in \mathbb{N}_0\}$ in each step is bounded by $\mathrm{E}(X_t - X_{t+1}|X_t = s) \geq \delta X_t$ where $1/\delta = \mathcal{O}(\sigma^2 K\sqrt{n})$. By Theorem 2, $\Pr(T > (\ln(X_0/x_{\min}) + \lambda)/\delta) \leq e^{-\lambda}$. Choosing $\lambda = d\ln n$ for any constant $d > 0$, the probability that $T = \Omega(K\sigma^2\sqrt{n}\log Kn)$ is at most n^{-d}.

Letting $\mathcal{E}$ be the event that the process has not failed by $\mathcal{O}(K\sigma^2\sqrt{n}\log Kn)$ steps, by the law of total probability, the hitting time of $X_t = 0$ (corresponding to the optimal allele frequency distribution) is bounded by $\mathcal{O}(K\sigma^2\sqrt{n}\log Kn)$ with probability $(1 - n^{-d})\Pr(\mathcal{E})$, where we can apply Lemma 5 to bound the probability of $\mathcal{E}$. ■

V. Experiments

A common way to handle noisy objective functions is to explicitly modify an existing algorithm to repeatedly resample the objective function value at each point and use the average of these values to estimate the true underlying objective function value. However, this can require many extra function evaluations to obtain a suitable estimate [31]. It has also been observed that even utilizing the expected information might not lead to robust solutions [27]. In this section, we seek to compare the performance of the cGA with a baseline hillclimber that uses explicit resampling to reduce the noise variance. Additionally, we also consider a cGA with explicit resampling (and fixed parameter K); however, after sufficient resampling, the landscape is deterministic and very

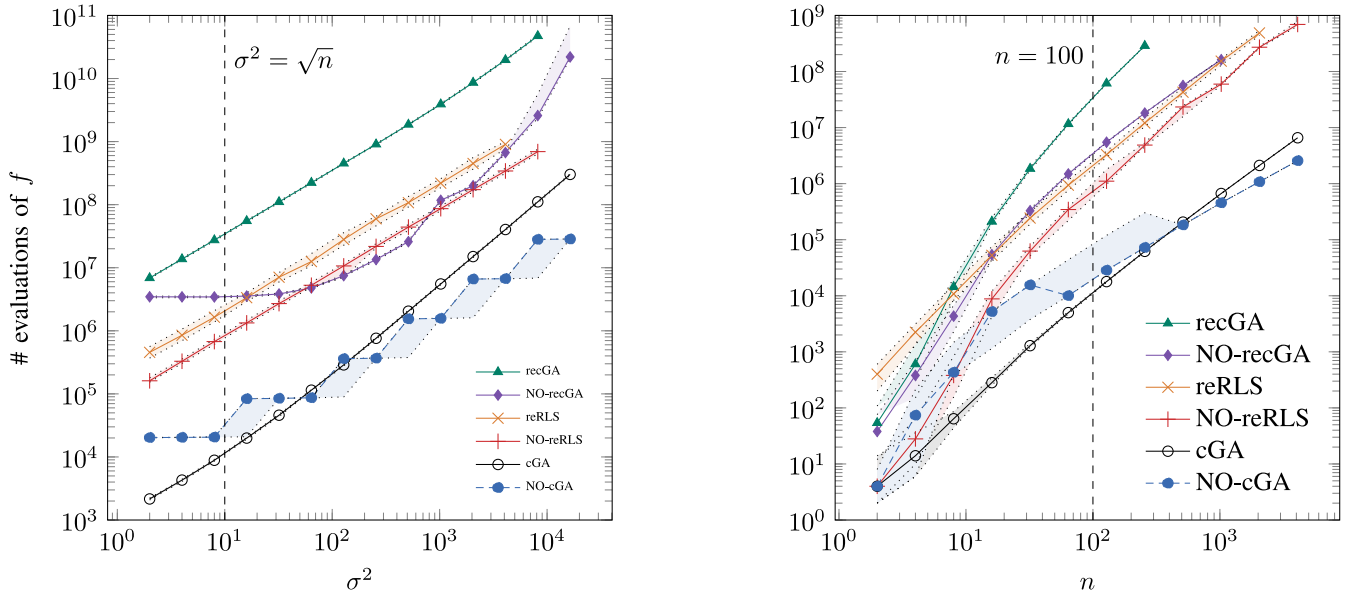

Fig. 1. Median run time as a function of noise variance for $n = 100$ (left) and as a function of n for $\sigma^2 = \sqrt{n}$ (right). One hundred runs at each point. Shaded area denotes interquartile range. This shows empirically that, for a wide range of noise intensities, the cGA outperforms a hillclimber with resampling by orders of magnitude. It also shows that it is better, again by orders of magnitude, to set K of the cGA according to the noise than to use a pure resampling strategy.

well-behaved, so that we expect the hill-climber to be superior in this case. Our baseline hillclimber is called resampling RLS (reRLS). For a particular variance σ^2, reRLS estimates the true objective function value by performing $20e \cdot \sigma^2 \ln n$ function calls to guarantee correct estimation of fitness throughout the optimization process (note that the term $\ln n$ is required to get truthful evaluations for poly(n) iterations). It then hill-climbs on the estimated true objective function by flipping a single bit chosen uniformly at random in each iteration and accepting points with equal or better estimated objectives. We also investigate a resampling variant of the cGA (recGA) that uses the same number of resamples as reRLS and has a noise-independent K. For the cGA, we use $K = 7\sigma^2\sqrt{n}(\ln n)^2$, and, for the recGA, we use $K = 7\sqrt{n}(\ln n)^2$, as we assume no noise after resampling.

All three algorithms require knowledge of the true noise variance to collect enough samples (reRLS and recGA) or to set K properly (cGA). We also investigate the performance of these approaches in the corresponding noise-oblivious setting as defined in Section II (NO-reRLS, NO-recGA, and NO-cGA). For the NO-reRLS, we use $T_\delta(\sigma^2) = n \ln n$, i.e., the run time bound is independent of σ^2 because we use resampling. However, we adjust the number of resamples, $20e \cdot \sigma^2 \ln n$, with the variance of the current run. For the NO-recGA, we also use $20e \cdot \sigma^2 \ln n$ resamples with respect to the current guessed variance, but we adjust $T_\delta(\sigma^2) = K\sigma^2\sqrt{n} \ln Kn$. For the NO-cGA, we use the same T_δ as for the NO-recGA, but we now also change $K = 7\sigma^2\sqrt{n}(\ln n)^2$ with respect to the current variance.

We measure the performance of reRLS and NO-reRLS by the number of calls to the objective function until the true optimum 1^n is generated. For the cGA, the NO-cGA, the recGA, and the NO-recGA, we count the number of fitness evaluations until the frequency vector p converges to the all-ones vector. This performance metric is standard within black-box optimization because, typically, objective function evaluation is the most costly operation in terms of computation time. For the cGA, this is twice the number of iterations through the while loop in Algorithm 3. For reRLS and the recGA, this is twice the number of iterations times the number of resamples necessary to obtain a suitable estimate of the true objective value.

The performance of each algorithm is plotted fixing $n = 100$ and controlling the variance in Fig. 1 (left). For each procedure and variance value, we run each algorithm 100 times until the true optimum is found and collect the number of calls to the objective function for each run. The median run times and their interquartile ranges are plotted. We also plot the performance as a function of n (fixing $\sigma^2 = \sqrt{n}$) in Fig. 1 (right).

Both results are plotted on log-log plots; thus, we see that the cGA and the NO-cGA are an order of magnitude faster than the algorithms using resampling in the noise ranges depicted; note that the noise range extends also to noise values much higher than any difference in objective value. We can also see that the noise-oblivious algorithms are faster than their counterparts that know the noise, since this knowledge results in setting the parameters in a way that guarantees success, while the noise-oblivious algorithms can use parameters that result in a lower run time with a lower success probability (and only increase parameters when necessary). Furthermore, we see that the recGA is worse than even the reRLS, which was to be expected, since both of them implicitly work on the deterministic OneMax function.

Fig. 2 corresponds to Fig. 1 and depicts the number of resamplings [(NO-)reRLS and (NO-)recGA] per iteration or the value of K [(NO-)cGA] that was sufficient for the respective algorithm to succeed. Note that the functions for the non-noise-oblivious algorithms have deterministic function

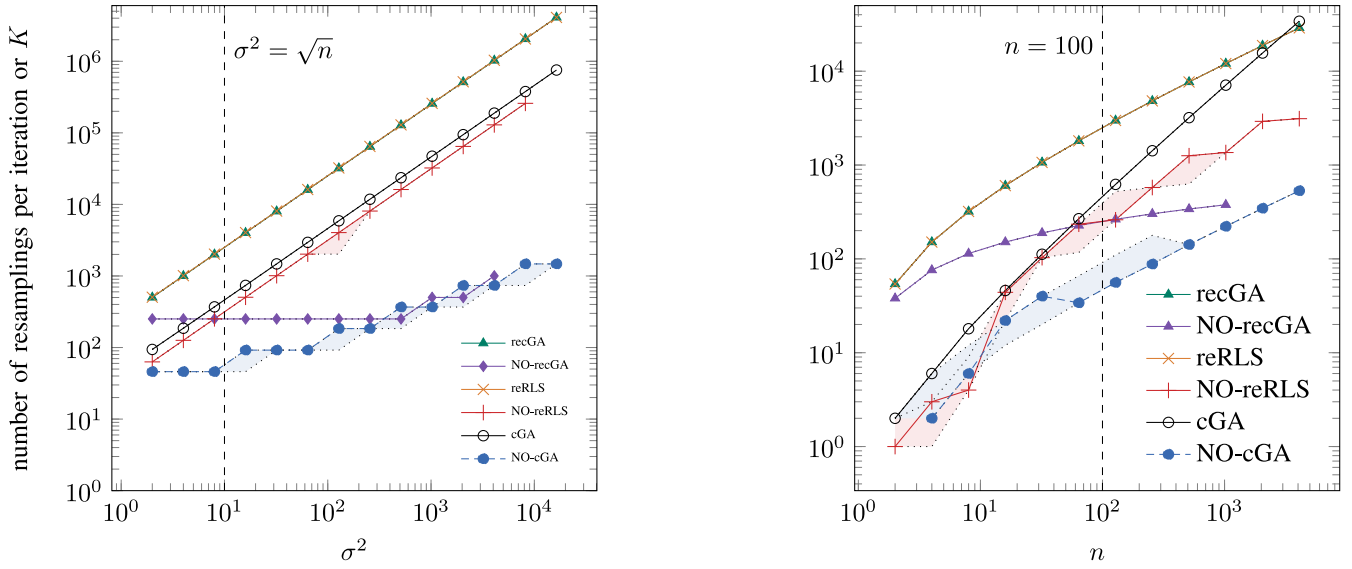

Fig. 2. Median of number of resamplings per iteration or median of K as a function of noise variance for $n = 100$ (left) or as a function of n for $\sigma^2 = \sqrt{n}$ (right), 100 runs at each point. Shaded area denotes interquartile range.

values, since the algorithms compute their respective parameters (number of resamplings or K), whereas the ones for the noise-oblivious versions are random variables because these algorithms vary their parameters according to the guessed variance so far.

Fig. 2 also indicates what a reasonable parameter value could look like, compared to the pessimistic (static) parameters of the non-noise-oblivious algorithms. One can see that the parameters of the noise-oblivious algorithms are always better than the ones of their counterparts. This means that the bounds on these parameters are not tight, as the values that suffice for optimization are orders of magnitudes below the static ones.

VI. Conclusion

In this paper, we examined the robustness of two approaches to noise, on the one hand an evolutionary algorithm [the $(\mu + 1)$ EA], on the other hand an EDA (the cGA). We saw that the cGA can handle noise efficiently by adjusting its step size to be more careful when the noise is large (Theorem 5). This we call *graceful scaling* with noise. Intuitively, the cGA can use its probability vector to average out the noise over many iterations.

We also saw that the $(\mu + 1)$ EA does not scale gracefully, i.e., for large noise, even large values of μ cannot cope with the noise (Corollary 1). The intuitive reason for this is that the probability of generating and accepting a worse individual becomes larger than the probability of generating and accepting a better individual: mutation has a bias toward bit strings with about as many 0s as 1s, and for high noise the probability of accepting slightly worse individuals is about $1/2$.

With our experimental results, we indicate that the robustness of the cGA is not just a theoretical result with possibly unrealistic constants. In fact, we showed that the cGA can handle noise even better than RLS (a straightforward hillclimber) with sufficient resampling, and we showed that increasing K is more efficient than pure resampling when running a cGA.

Our results highlight the importance of understanding the influence of different search operators in uncertain environments and suggest that algorithms, such as the cGA, are able to scale gracefully with noise.

While all our results considered first hitting times, one can also look at these problems from the perspective of a fixed time budget [22]. We believe that similar results can be obtained in this setting.

Appendix

In this section, we give a detailed derivation of the solution to the definite integral needed in Lemma 3.

Lemma 6: Let $k \in \mathbb{N}$ and $t \in \mathbb{R}$, then

$$\frac{1}{\pi}\int_{-\infty}^{\infty} \frac{1-\cos^k(t)}{t^2}\,\mathrm{d}t = 2\lceil k/2\rceil \binom{2\lceil k/2\rceil}{\lceil k/2\rceil} 2^{-2\lceil k/2\rceil}.$$

Proof: We express the kth power of $\cos(t)$ in terms of the binomial expansion

$$\cos^k(t) = \left(\frac{e^{it}+e^{-it}}{2}\right)^k = \frac{1}{2^k}\sum_{j=0}^{k}\binom{k}{j}\cos((2j-k)t).$$

Thus, we may calculate the indefinite integral

$$\begin{aligned}\int \frac{1-\cos^k(t)}{t^2}\,\mathrm{d}t &= -\frac{1}{2^k}\sum_{j=0}^{k}\binom{k}{j}\int\frac{\cos((2j-k)t)}{t^2}\,\mathrm{d}t - \frac{1}{t}\\ &= \frac{\cos^k(t)}{t} - \frac{1}{2^k}\sum_{j=0}^{k}\binom{k}{j}(k-2j)\,\mathrm{Si}((2j-k)t) - \frac{1}{t} \qquad (7)\end{aligned}$$

where $\mathrm{Si}(z) \coloneqq \int_0^z (\sin x/x)\,\mathrm{d}x$ is the sine integral. We define the function

$$h_k(t) \coloneqq -\frac{1}{2^k}\sum_{j=0}^{k}\binom{k}{j}(k-2j)\,\mathrm{Si}((2j-k)t).$$

By the algebraic limit theorem

$$\lim_{t\to\infty} h_k(t) = -\frac{1}{2^k}\sum_{j=0}^{k}\binom{k}{j}(k-2j)\lim_{t\to\infty}\mathrm{Si}((2j-k)t).$$

The limits at infinity of the sine integral are

$$\lim_{t\to\infty}\mathrm{Si}((2j-k)t) = \begin{cases}\pi/2 & \text{if } (2j-k)>0\\ 0 & \text{if } (2j-k)=0\\ -\pi/2 & \text{if } (2j-k)<0.\end{cases}$$

Therefore

$$\begin{aligned}\lim_{t\to\infty} h_k(t) &= \frac{-\pi}{2^{k+1}}\left(\sum_{j=\lceil k/2\rceil}^{k}\binom{k}{j}(k-2j) - \sum_{j=0}^{\lfloor k/2\rfloor}\binom{k}{j}(k-2j)\right)\\ &= \frac{\pi}{2^{k+1}}\left(\sum_{j=0}^{\lfloor k/2\rfloor}\binom{k}{j}(k-2j) - \sum_{j=\lceil k/2\rceil}^{k}\binom{k}{j}(k-2j)\right)\\ &= \frac{\pi}{2^{k+1}}\left(\lceil k/2\rceil\binom{2\lceil k/2\rceil}{\lceil k/2\rceil}I(k)\right)\end{aligned}$$

where

$$I(k) = \begin{cases}1 & \text{if } k \text{ is odd}\\ 2 & \text{if } k \text{ is even.}\end{cases}$$

It is possible to express $I(k) = 2^{(k+1)}/2^{2\lceil k/2\rceil}$, and so

$$\lim_{t\to\infty} h_k(t) = \pi\lceil k/2\rceil\binom{2\lceil k/2\rceil}{\lceil k/2\rceil}2^{-2\lceil k/2\rceil}. \tag{8}$$

A similar derivation yields

$$\lim_{t\to-\infty} h_k(t) = -\lim_{t\to\infty} h_k(t). \tag{9}$$

To complete the proof, we take the indefinite integral derived in (7) and use the fact that $\lim_{t\to\infty}\cos^k(t)/t = \lim_{t\to\infty} 1/t = 0$ to compute the definite integral as follows:

$$\begin{aligned}\int_{-\infty}^{\infty}\frac{1-\cos^k(t)}{t^2}\,\mathrm{d}t &= \lim_{t\to\infty}\left(\frac{\cos^k(t)}{t} + h_k(t) - \tfrac{1}{t}\right)\\ &\quad - \lim_{t\to-\infty}\left(\frac{\cos^k(t)}{t} + h_k(t) - \tfrac{1}{t}\right)\\ &= \lim_{t\to\infty} h_k(t) - \lim_{t\to-\infty} h_k(t) = 2\lim_{t\to\infty} h_k(t)\end{aligned}$$

where we have used (9). Finally, substituting (8) for the limit at positive infinity, we get

$$= \pi\cdot 2\lceil k/2\rceil\binom{2\lceil k/2\rceil}{\lceil k/2\rceil}2^{-2\lceil k/2\rceil}. \qquad \blacksquare$$

REFERENCES

[1] Y. Akimoto, S. Astete-Morales, and O. Teytaud, "Analysis of runtime of optimization algorithms for noisy functions over discrete codomains," *Theor. Comput. Sci.*, vol. 605, pp. 42–50, Nov. 2015.

[2] D. V. Arnold and H.-G. Beyer, *Efficiency and Mutation Strength Adaptation of the* $(\mu/\mu_I, \lambda)$*-ES in a Noisy Environment*. Heidelberg, Germany: Springer, 2000, pp. 39–48.

[3] H.-G. Beyer, "Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice," *Comput. Methods Appl. Mech. Eng.*, vol. 186, nos. 2–4, pp. 239–267, 2000.

[4] H.-G. Beyer and B. Sendhoff, "Robust optimization—A comprehensive survey," *Comput. Methods Appl. Mech. Eng.*, vol. 196, nos. 33–34, pp. 3190–3218, 2007.

[5] H.-G. Beyer and B. Sendhoff, "Evolutionary algorithms in the presence of noise: To sample or not to sample," in *Proc. 1st IEEE Symp. Found. Comput. Intell. (FOCI)*, Honolulu, HI, USA, 2007, pp. 17–24.

[6] H.-G. Beyer, M. Olhofer, and B. Sendhoff, "On the behavior of $(\mu/\mu_I, \lambda)$-ES optimizing functions disturbed by generalized noise," in *Foundations of Genetic Algorithms, 7*. San Francisco, CA, USA: Morgan Kaufmann, 2003, pp. 307–328.

[7] J. Branke and C. Schmidt, "Selection in the presence of noise," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Chicago, IL, USA, 2003, pp. 766–777.

[8] J. Branke and C. Schmidt, "Sequential sampling in noisy environments," in *Proc. Parallel Problem Solving Nat. (PPSN)*, Birmingham, U.K., 2004, pp. 202–211.

[9] S.-H. Chang, P. C. Cosman, and L. B. Milstein, "Chernoff-type bounds for the Gaussian error function," *IEEE Trans. Commun.*, vol. 59, no. 11, pp. 2939–2944, Nov. 2011.

[10] B. Doerr, A. Hota, and T. Kötzing, "Ants easily solve stochastic shortest path problems," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Philadelphia, PA, USA, 2012, pp. 17–24.

[11] B. Doerr, "Analyzing randomized search heuristics: Tools from probability theory," in *Theory of Randomized Search Heuristics*, vol. 1. Singapore: World Sci., 2011, pp. 1–20.

[12] B. Doerr and L. A. Goldberg, "Adaptive drift analysis," *Algorithmica*, vol. 65, no. 1, pp. 224–250, 2013.

[13] S. Droste, "Analysis of the (1 + 1) EA for a Noisy OneMax," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Seattle, WA, USA, 2004, pp. 1088–1099.

[14] S. Droste, "A rigorous analysis of the compact genetic algorithm for linear functions," *Nat. Comput.*, vol. 5, no. 3, pp. 257–283, Aug. 2006.

[15] M. Feldmann and T. Kötzing, "Optimizing expected path lengths with ant colony optimization using fitness proportional update," in *Proc. Found. Genet. Algorithms (FOGA)*, Adelaide, SA, Australia, 2013, pp. 65–74.

[16] J. M. Fitzpatrick and J. J. Grefenstette, "Genetic algorithms in noisy environments," *Mach. Learn. Special Issue Genet. Algorithms*, vol. 3, no. 2, pp. 101–120, 1988.

[17] T. Friedrich, T. Kötzing, M. S. Krejca, and A. M. Sutton, "The benefit of recombination in noisy evolutionary search," in *Proc. Int. Symp. Algorithms Comput. (ISAAC)*, Nagoya, Japan, 2015, pp. 140–150.

[18] C. Gießen and T. Kötzing, "Robustness of populations in stochastic environments," *Algorithmica*, vol. 75, no. 3, pp. 462–489, 2016.

[19] W. J. Gutjahr and G. C. Pflug, "Simulated annealing for noisy cost functions," *J. Glob. Optim.*, vol. 8, no. 1, pp. 1–13, 1996.

[20] N. Hansen, "Benchmarking a BI-population CMA-ES on the BBOB-2009 noisy testbed," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Montreal, QC, Canada, 2009, pp. 2397–2402.

[21] G. R. Harik, F. G. Lobo, and D. E. Goldberg, "The compact genetic algorithm," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 287–297, Nov. 1999.

[22] T. Jansen and C. Zarges, "Performance analysis of randomised search heuristics operating with a fixed budget," *Theor. Comput. Sci.*, vol. 545, pp. 39–58, Aug. 2014.

[23] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments—A survey," *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 303–317, Jun. 2005.

[24] T. Koshy, *Catalan Numbers With Applications*. New York, NY, USA: Oxford Univ. Press, 2009.

[25] T. Kötzing, "Concentration of first hitting times under additive drift," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Vancouver, BC, Canada, 2014, pp. 1391–1398.

[26] P. K. Lehre and C. Witt, "Concentrated hitting times of randomized search heuristics with variable drift," in *Proc. Int. Symp. Algorithms Comput. (ISAAC)*, Jeonju, South Korea, 2014, pp. 686–697.

[27] Y. Mei, K. Tang, and X. Yao, "Capacitated arc routing problem in uncertain environments," in *Proc. Congr. Evol. Comput. (CEC)*, Barcelona, Spain, 2010, pp. 1–8.
[28] D. S. Mitrinović, *Analytic Inequalities* (Grundlehren der Mathematischen Wissenschaften). Heidelberg, Germany: Springer, 1970.
[29] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 1995.
[30] A. Prügel-Bennett, J. E. Rowe, and J. Shapiro, "Run-time analysis of population-based evolutionary algorithm in noisy environments," in *Proc. Found. Genet. Algorithms (FOGA)*, Aberystwyth, U.K., 2015, pp. 69–75.
[31] C. Qian, Y. Yu, Y. Jin, and Z.-H. Zhou, "On the effectiveness of sampling for evolutionary optimization in noisy environments," in *Proc. Parallel Problem Solving Nat. (PPSN)*, Ljubljana, Slovenia, 2014, pp. 302–311.
[32] J. Spanier and K. B. Oldham, *An Atlas of Functions*. Washington, DC, USA: Hemisphere, 1987.
[33] D. Sudholt and C. Thyssen, "A simple ant colony optimizer for stochastic shortest path problems," *Algorithmica*, vol. 64, no. 4, pp. 643–672, 2012.
[34] B. von Bahr and C.-G. Esseen, "Inequalities for the r-th absolute moment of a sum of random variables, $1 \leq r \leq 2$," *Ann. Math. Stat.*, vol. 36, no. 1, pp. 299–303, 1965.

Tobias Friedrich received the M.S. degree in computer science from the University of Sheffield, Sheffield, U.K., in 2003, the Diploma degree in mathematics from the University of Jena, Jena, Germany, in 2005, and the Ph.D. degree in computer science from Saarland University, Saarbrücken, Germany, in 2007.

He was a Post-Doctoral Fellow with the Algorithms Group, International Computer Science Institute, Berkley, CA, USA. From 2011 to 2012, he was a Senior Researcher with the Max Planck Institute for Informatics, Saarbrücken, and an Independent Research Group Leader with the Cluster of Excellence on Multimodal Computing and Interaction, Saarbrücken. From 2012 to 2015, he was a Full Professor and the Chair of theoretical computer science with the University of Jena. Since 2015, he has been a Full Professor with the University of Potsdam, Potsdam, Germany, and the Head of the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam. His current research interests include randomized methods in mathematics and computer science and randomized algorithms (both classical and evolutionary).

Timo Kötzing received the Ph.D. degree in computer science from the University of Delaware, Newark, DE, USA, in 2009.

He was a Post-Doctoral Fellow with the Algorithms and Complexity Group, Max Planck Institute for Informatics, Saarbrücken, Germany. From 2013 to 2015, he was a Post-Doctoral Researcher with the University of Jena, Jena, Germany. Since 2015, he has been a Post-Doctoral Researcher with the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam, Germany. His current research interests include theoretical foundation of learning theory as well as the analysis of randomized search heuristics and the development of efficient tools for this goal.

Martin S. Krejca received the B.S. and M.S. degrees in computer science from the University of Jena, Jena, Germany, in 2012 and 2014, respectively. He is currently pursuing the Ph.D. degree with the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam, Germany.

His current research interests include the theoretical analysis of evolutionary algorithms, especially the analysis of estimation of distribution algorithms.

Andrew M. Sutton received the M.S. and Ph.D. degrees in computer science from Colorado State University, Fort Collins, CO, USA, in 2006 and 2011, respectively.

He has held Post-Doctoral Research fellowships with the University of Adelaide, Adelaide, SA, Australia, and the University of Jena, Jena, Germany. He is currently a Researcher with the Algorithm Engineering Group, Hasso Plattner Institute, Potsdam, Germany. His current research interest includes the theoretical analysis of randomized search heuristics.