Experimental Analysis of Evolution Strategies – Overview and Comprehensive Introduction

Thomas Bartz–Beielstein

Universtität Dortmund, D-44221 Dortmund, Germany
Thomas.Beielstein@udo.edu,
WWW home page: http://ls11-www.cs.uni-dortmund.de/people/tom/index.html

Abstract. This article presents statistical techniques for the design and analysis of evolution strategies. These techniques can be applied to other search heuristics such as genetic algorithms, simulated annealing or particle swarm optimizers. It provides guidelines for the comparison of different algorithms on artifical test functions and on real-world optimization problems. Statistical experimental design techniques to improve the integrity and comparability of experiments are proposed. Interpreting the run of an optimization algorithm as an experiment, design of experiments (DOE), response surface methods (RSM), and tree-based regression methods can be applied to analyze and to improve its performance. We recommmend to base the comparison of algorithms on "tuned" algorithms and not on their "standard" parameterizations.

1 Introduction

At present, it is intensely discussed which type of experimental research methodologies should be used to improve the acceptance and quality of evolutionary algorithms (EA). A broad spectrum of presentation techniques makes new results in evolutionary computation (EC) almost incomparable. Discussions and sessions related to this subject took part during the congress on evolutionary computation (CEC) and on the genetic and evolutionary computation conference (GECCO). I.e., how to promote good standards and quality of research in the field of EC was discussed during the international society for genetic and evolutionary computation (ISGEC) workshop on standards at GECCO in 2002 [1]. Eiben and Jelasity (2002) explicitely list four problems resulting from this situation:

- the lack of standardized test-functions, or benchmark problems,
- the usage of different performance measures,
- the impreciseness of results, and therefore no clearly specified conclusions, and
- the lack of reproducibility of experiments.

EC shares these problems with other scientific disciplines [3]. Solutions from these other disciplines, that have been successfully applied for many years, might

Table 1. Evolution strategy compared to simulated annealing. First runs with "standard" parameterizations discussed later on. Best value from 5,000 iterations. The object variables x_i were initialized as shown in Eq. 4.

Problem	Dimension	Global optimum	ES	SANN
Sphere	12	0	$\mathbf{1.378548e-36}$	$9.936222e-08$
Rosenbrock	12	0	0.02246728	**0.0001481394**

be transferable to EC. Here we can mention: statistical design of experiments (DOE) [4], design of computational experiments to test heuristics [5, 6], experimental algorithmics [7], experimental designs for simulation [8], or deterministic computer experiments [9].

In this chapter, we first start with a comparison of two stochastic global optimization methods. This comparison is based on real optimization data, but it is kept as simple as possible for didactical purposes. An evolution strategy is compared to simulated annealing (SANN) [10, 11]. The first experiments are based on the "standard" parameterizations found in the literature [11, 12]. These parameterizations will be discussed later on. The results presented in Tab. 1 might lead to the conclusion that the ES outperforms the SANN on the sphere function whereas the SANN method works better on Rosenbrock's "banana" function. Modifying the exogenous parameters of the ES, i.e. replacing the comma selection with the plus selection and reducing the initial step-size from 3.0 to 0.3 leads to a different result: the best value found now reads $5.027801e-08$. This might lead to the conclusion that ES perform better than SANN in *any* case. However, it is still an open question what happens if the parameter values of the SANN are improved or if repeat optimization runs are performed. But before we start performing repeat runs, we should determine how many repeats are necessary to obtain significant results.
The remainder of this article[1] deals with questions related to these problems and provides a methodology to perform comparisons in a statistically sound manner.

Optimization runs can be regarded as experiments. In our approach, an experiment consists of a problem and its related fitness function, an algorithm, and a quality criterion: we will use design of experiments, regression analysis, and generalized linear models, to improve and compare algorithms' performances. The main focus in this paper lies on natural problem classes: its elements are problems that are based on real-world optimization problems in contrast to artificial problem classes [2].

The approach presented here is based on DOE methods and can be used to solve the following tasks:

Investigation: Analyzing and tuning models and optimization criteria: i.e. what are important parameters, what should be optimized?

[1] This article is an extended and updated version of [13].

Comparison: Comparing the performance of competing search heuristics such as evolutionary algorithms, simulated annealing, particle swarm optimization etc.

Conjecture: It might be good to demonstrate performance, but it is better to explain performance. Understanding and further research, based on statistics and visualization techniques, play important roles in the approach presented in this paper. Cohen et al. (2000) mention some reasons why experiments should complement the theoretical analysis: theories may have holes, observations may suggest new theories and theories can be tested by experiments (see also Fig. 1). Gregory et al. (1996) performed an interesting study of dynamic scheduling that demonstrates how synergetic effects between experiment and theory can evolve. The methodology presented in their study is closely related to our approach.

Quality: Improving the robustness of the results obtained in optimization or simulation runs. Robustness includes insensitivity to exogenous factors that can affect algorithms' performance, and minimization of the variability around the obtained solutions [16].

Hence, the approach presented here might be interesting for optimization practitioners who are confronted with a complex real-world optimization problem in a situation where only few preliminary investigations are possible to find good parameter settings [17, 18]. Kleijnen and Pala (1999) describe a competition, organized by the Business Section of the Netherlands Society for Statistics and Operation Research: only 32 runs are permitted to obtain the maximum output for a given simulation model by selecting the best combination of six inputs. Furthermore, DOE is applicable a priori to tune different parameter settings of two algorithms to provide a fair comparison. Additionally, these methods can be used in other contexts to improve the optimization runs, i. e. to generate systematically feasible starting points that are better than randomly generated initial points.

Our approach differs from the following eight approaches:

1. Classical design of experiments as used in industrial optimization. We will consider "optimization via simulation": a simulation model equipped with a fitness function defines an optimization problem. Therefore, simulation and optimization will be used equivalently throughout the remainder of this article. DOE techniques must be adapted if applied to simulation models since stochastic simulation uses pseudo-random numbers. *Randomness is replaced by pseudo-randomness.* As common or antithetic seeds can be used, the simulation practitioner has much more control over the noise in the experiments and can control the source of variability [18]. The different simulation runs for one specific factor combination can be performed under exactly the same conditions. Blocking and randomization, important techniques to reduce the systematic influence of different experimental conditions, are unnecessary in computer-based simulation. The random number seed is the only random element in a random simulation model.

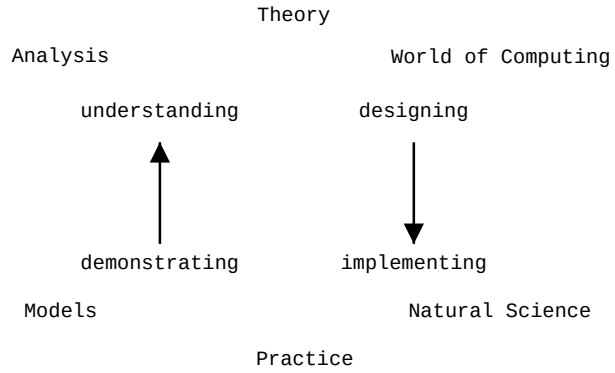

Fig. 1. Relationship between theory and practice. Practice can benefit from theory and vice versa. Demonstrating good results is only the first step in the scientific process, whereas nature's reality can be seen as the judge of a scientific theory.

2. Meta-EA approaches, see, e.g., [12, 20]. Meta-algorithms might locate good parameter sets, though without providing much insight as how sensitive performance is to parameter changes.
3. Schaffer's study of control parameters of GA [21]. Schaffer proposes a complete factorial design experiment that requires $8,400$ run configurations, each configuration was run to $10,000$ fitness function evaluations. In contrast to this, we propose an approach that requires a small amount of fitness function evaluations only. Furthermore, our approach takes the underlying problem instance into account and does not draw any conclusions that are problem independent.
4. Parameter control as introduced in [22]. Parameter control deals with parameter values that are changed during the optimization run. This is in contrast to our approach that is based on parameter values that are specified before the run is performed. The assumption that specific problems require specific EA parameter settings is common to both approaches [23].
5. The exemplary study of simulated annealing by David Johnson's group [24, 25]. Regarding this study, our approach is related to the discipline experimental algorithmics [26]. Thus it can be added to the category "assessment of heuristics" as classified in [7]. Experimental algorithmics offers methodologies for the design, implementation, and performance analysis of computer programs for solving algorithmic problems. Different from this approach, our goal is to provide methods for very complex real-world problems, when only a few optimization runs are possible, i.e. optimization via simulation. The elevator supervisory group controller study discussed in [27] required more than a full week of round-the-clock computing in a batch job processing system to test 80 configurations.
6. Empirical modeling of genetic algorithms as presented by [28]. Their methodology has a different goal than our approach: we are trying to tune an evo-

lutionary algorithm with the fewest amount of experiments, whereas Myers' and Hancock's approach requires 129,600 program runs.
7. François' and Lavergne's statistical approach [29]. They demonstrate the applicability of generalized linear models to design evolutionary algorithms. This approach is similar to [28]. Again, data sets of size 1,000 or even more are necessary, although a simplified evolutionary algorithm with two parameters only (`Moses`) is designed.
8. Design and analysis of computer experiments as introduced in [9], where the deterministic output of a computer experiment is modeled as the realization of a stochastic process. This approach differs significantly from our approach, the same applies to design and analysis of computer experiment (DACE) methods [30].

Despite the differences mentioned above, it might be beneficial to adapt some of these well-established ideas from other fields of research to improve the acceptance and quality of evolutionary algorithms.

Although the no free lunch theorem (NFL) for search states that there does not exist any algorithm that is better than another over all possible instances of optimization problems, this result does not imply that we should not compare different algorithms. Keeping in mind that we are considering problems of practical interest, the reader may refer to the discussion in [31–33].

Hence, the main contribution of this paper is to propose an answer to the complex questions: how to determine strategy parameters for search algorithms that are suitable to optimize efficiently and effectively real–world optimization problems? And, how can two different optimization algorithms be compared in a statistically sound manner? Thus, we can provide at least partial answers to the problems mentioned by Eiben and Jelasity. These answers will be given from the view-point of an optimization practitioner.

Drawing conclusions from the experimental data is an important part of the approach presented here. This approach will be referred to as the "experimental analysis of search heuristics" throughout the rest of this paper.

This paper is structured as follows: section 2 describes how evolution strategies (ES), as a special class of evolutionary algorithms, can be parameterized [10]. Hence, the run of an ES can be treated as an experiment. Different measures to compare stochastic optimizers for complex real-world problems are discussed in Sec. 3. Section 4 is devoted to DOE methods that are used to tune and compare different optimization run configurations. Comprehensive examples for tuning and comparison are presented in Sec. 5. Section 6 discusses the limits of our approach. A summary and an outlook are given in Sec. 7.

2 Evolution Strategies

2.1 Evolutionary Algorithms as Experiments

The popularity of evolutionary algorithms has steadily increased in the last four decades. There are many degrees of freedom when starting an optimization run: as other search algorithms such as simulated annealing [34] or particle

swarm optimization [35, 36], an EA requires the determination of parameters such as the population size before the optimization run is performed. The optimization of real-world problems requires good initial parameters, since many real-world problems are computationally expensive, e.g., "optimization via simulation" [37–39]. Therefore only a few optimization runs are possible, that should be performed with "good" parameter settings. Optimization practitioners are interested in determing a good parameter setting with a minimum amount of optimization or simulation runs. The choice of an adequate parameter setting, or EA design, can be based on expert knowledge. But in many cases there is no such knowlegde available.

DOE techniques can be applied to optimization algorithms such as evolutionary algorithms, considering the run of an algorithm as an experiment, gaining insightful conclusions into the behavior of the algorithm and the interaction and significance of its parameters. The first two steps in this analysis are called screening and modeling. In a third step, we can use response surface methods (RSM) to improve significant parameter settings (optimization) [16]. Therefore, DOE methods, combined with response surface methods, might lead to an improved EA design.

From the viewpoint of an experimenter, factors can be defined as parameters, variables, and behavioral relationships, that can be changed during an experiment. How factors can be interpreted from a statistical point of view will be discussed in Sec. 4. Generally, there are two different types of factors that influence the behavior of an optimization algorithm: problem specific factors, i.e. the fitness function, and algorithm specific factors, i.e. the population size. The latter can be divided into endogenous and exogenous factors. Exogenous factors, or "exogenous strategy parameters" as they are called in [10], are kept constant during the optimization run, whereas endogenous strategy parameters, i.e. standard deviations[2], are modified by the algorithms during the run.

2.2 Evolution Strategies: Algorithm Specific Factors

The methodology presented in this paper leads to results that are tailored for one specific algorithm–optimization problem combination. In a similiar manner as [40] mention the nonsense of speaking of a problem complexity without considering the parameterization of the optimization algorithm, we cannot discuss the behavior of an algorithm without taking the underlying problem into account. Our experimental design approach is transferable to any kind of EA design or even any parameterizable stochastic search algorithm such as simulated annealing, tabu search or genetic algorithms. To give an example, we will analyze evolution strategies.

Beyer and Schwefel (2002) provide a comprehensive introduction to this special class of EA. An indepth discussion of evolutionary algorithms and other direct search methods can be found in Schwefel's seminal book "Evolution and

[2] Standard deviations will be laxly referred to as "step-width" or "mutation strenghts".

Table 2. Default settings of exogenous parameters of a "standard" evolution strategy. From [12]. Bäck does not recommend to use this "standard" without reflection. He additionally presents a kind of *default hierarchy* that includes four parameterizations for simple and complex algorithms and suggests to perform experiments. Hence, our approach can be seen as an extension of Bäck's methods. Problems may occur, when these "standards" are blindly adopted and not adjusted to the specific optimization problem.

Symbol	Factor	Parameter	Range	Default Values
μ	P	Number of parent individuals	$\mathbb{N}$	15
$\nu = \lambda/\mu$	S	Offspring-parent ratio	$\mathbb{R}_+$	7
$\sigma_i^{(0)}$	InitS	Initial standard deviations	$\mathbb{R}_+$	3
n_σ	NSigma	Number of standard deviations. D denotes the problem dimension	$\{1, D\}$	D
c_τ	TauMult	Multiplier for individual and global mutation parameters	$\mathbb{R}_+$	1
ρ	Rho	Mixing number	$\{1, \mu\}$	μ
r_x	XReco	Recombination operator for object variables	$\{i, d\}$	d (discrete)
r_σ	SReco	Recombination operator for strategy variables	$\{i, d\}$	i (intermediate)
κ	K	Maximum age	$\mathbb{R}_+$	1

Optimum Seeking" from 1995. This book is a slightly extended version of Schwefel's PhD thesis from 1975 that was published under the title "Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie"[43], and translated into English four years later [44].

An ES-algorithm run can be described briefly as follows: the parental population is initialized at time (generation) $t = 0$. Then λ offspring individuals are generated in the following manner: a parent family of size ρ is selected randomly from the parent population. Recombination is applied to the object variables and the strategy parameters. The mutation operator is applied to the resulting offspring vector. Selection is performed to generate the next parent population. A termination criterion is tested. If this criterion is not fulfilled, the generation counter (t) is incremented and the process continues with the generation of the next offspring.

In the following, we extend the classical $(\mu/\rho \stackrel{+}{,} \lambda)$ ES notation to an experimental design vector representation. We consider the parameters or control variables mentioned below:

1. Number of parent individuals: μ.
2. Number of offspring individuals: λ. Based on μ and λ, the selection pressure ν is defined as the offspring–parent ratio λ/μ. For given μ and ν values, λ is calculated as $\mu \cdot \nu$ and rounded to the nearest whole number.
3. Initial mean step sizes (standard deviations of the mutations of the decision variables): $\sigma_i^{(0)}$, $i = 1, \ldots, n_\sigma$.
   The algorithms' performance may increase if problem specific $\sigma_i^{(0)}$ values for

each dimension are chosen. To prevent an exponential blow up in the number of ES parameterizations, we assume scaled object variables. Therefore, only one initial step size is necessary for all dimensions: $\sigma^{(0)} = \sigma_i^{(0)} \quad \forall\, i \in \{1, \ldots, D\}$. The relevance of this parameter decreases with an increasing number of permitted iteration steps. Unfortunately, many real-world optimization problems permit only a small number of iterations. Therefore, the selection of an adequate $\sigma^{(0)}$ value might improve the performance.

4. Number of standard deviations: n_σ with $1 \leq n_\sigma \leq D$. D denotes the problem dimension.
5. Multiplier for the individual mutation (learning) parameter and the global mutation parameter: c_τ.
   The law of large numbers implies that the average overall step size of a parent and of its offspring do not differ very much for $D \gg 1$. Introducing an overall step size (global mutation parameter) τ_0 and individual step sizes (individual mutation parameter) τ might avoid this effect. The value c_τ is used as a scaling parameter for τ_0 and τ:

   $$\tau_0 = c_\tau / \sqrt{2D} \quad \text{and} \quad \tau = c_\tau / \sqrt{2\sqrt{D}}.$$

   Mutation is based on the extended log-normal rule, that enables learning perpendicular mutation ellipsoids [10]:

   $$\sigma^{(t+1)} = \exp(\tau_0 \mathcal{N}_0(0,1)) \cdot \left(\sigma_1^{(t)} \exp(\tau \mathcal{N}_1(0,1)), \ldots, \sigma_D^{(t)} \exp(\tau \mathcal{N}_D(0,1))\right),$$

   where $\sigma^{(t)}$ denotes the step length vector[3] at iteration t, and $\mathcal{N}(0,1)$ is the realization of a normally distributed random number with variance one and expectation zero. Although $c_\tau = 1$ leads to "standard" values that can be found in the literature [45]: $\tau_0 = 1/\sqrt{2D}$ and $\tau = 1/\sqrt{2\sqrt{D}}$, we cannot recommend this parameterization. I.e., high dimensional fitness functions might require completely different τ resp. τ_0 values.
6. Mixing number: ρ. The mixing number denotes the size of the parent family that is chosen from the parent pool of size μ to create one offspring.
7. Recombination operator for object variables: r_x.
   There exists a broad variety of recombination operators. Although [12] already describes 7 different types: no recombination, discrete (dominant), panmictic discrete, intermediate, panmictic intermediate, generalized intermediate, and panmictic generalized intermediate, this listing is by far not complete.[4] Beyer (1996) calculates the global arithmetic mean (center of mass of the ρ parent vectors) in contrast to other recombination schemes that determine the arithmetic average of 2 parent vector coordinates chosen out of ρ [12, 20]. Our analysis takes Beyer's recombination scheme from 1996

[3] For the sake of simplicity, we will use no arrows to denote vectors in this article. It will generally be clear from context whether a variable is a vector or a scalar quantity.

[4] I.e., it does not include the recombination operator that was used in [42].

into account, since it is used in a recent publication [10]. Therefore, we consider discrete (d) and intermediate (i) recombination methods that depend on the mixing number ρ. The mixing number is treated as a qualitative factor with two levels: b denotes the bisexual scheme with $\rho = 2$, whereas m is the multisexual recombination scheme with $\rho = \mu$.
The discrete recombination scheme is often recommended as a rule of thumb for the object variables. We do not recommend this, but suggest to chose the recombination operator for each specific optimization problem anew. This is in accordance with empirical results found by Kursawe, but in contradiction to theoretical results from [46].

8. Recombination operator for strategy variables: r_σ.
9. Selection mechanism, maximum life span: κ.
   Plus-strategies $(\mu + \lambda)$, and comma-strategies (μ, λ) can be generalized by introducing the parameter κ that defines the maximum age (in generations) of an individual. If κ is set to 1, we obtain the comma-strategy, if κ equals $+\infty$, we model the plus-strategy.[5]

The following vector notation provides a compact description of an ES parameter design [10, 47]:

$$p_{\text{ES}} = \left(\mu, \nu, \sigma^{(0)}, n_\sigma, c_\tau, \rho, r_x, r_\sigma, \kappa\right). \tag{1}$$

This representation will be used throughout the rest of this article and is summarized in Tab. 2. This table shows the capital letters that were used to denote the variables in the figures and in the regression models, and also typical parameter settings[6]. The "standard" parameterization from [12] reads:

$$p_{\text{STD}} = \left(\mu = 15, \nu = 7, \sigma^{(0)} = 3, n_\sigma = D, c_\tau = 1, \rho = \mu, r_x = d, r_\sigma = i, \kappa = 1\right). \tag{2}$$

Each ES-run requires the specification of at least one random seed r_0 (see Eq. 5). Experimental conditions in computer based optimization include the random variates that are used during the optimization run to generate new candidate solutions. Variance reduction techniques such as common random numbers (CRN) are based on the idea that alternative configurations should be compared under similar conditions. Thus, in the comparison of different run configurations the same random seed is used. CRN and other variance reduction techniques are discussed in detail in [48].

For each of the μ different parents at generation 0, a starting vector $x_j^{(0)}, \quad j = 1, \ldots, \mu$, has to be selected and a stop criterion has to be specified in addition to the parameters mentioned so far. These parameters are considered as problem and not algorithm specific in our approach.

[5] The symbol "∞", that represents a plus-strategy for the κ variable, is encoded as "-1".

[6] Bäck (1996) states that *the setting of $\sigma^{(0)}$ depends on the particular objective function. $\sigma^{(0)} = 3$, however, turns out to be a reasonable choice even if nothing is known about the objective function.*

A vector notation can be used for the optimization and problem specific parameters such as the fitness function f, the problem dimension D, the initial values for the variables to be optimized $x^{(0)} = (x_1^{(0)}, \ldots, x_D^{(0)})$, the number of iterations (function evaluations) N_{tot}, or the number of repeats for each scenario N_{exp}. The vector

$$d = (f, D, x^{(0)}, N_{\mathrm{tot}}, N_{\mathrm{exp}}, \ldots) \tag{3}$$

denotes the corresponding design configuration. The initial values $x_i^{(0)}$ were set as denoted in Eq. 4. An optimization run configuration consists of the problem specific design configuration, e.g. represented by d as shown in Eq. 3 and the parameter design configuration p, e.g. as shown in Eq. 1.

In the following sections, we will discuss how different algorithms can be compared. Based on these considerations we are able to apply DOE methods to improve the behavior of the ES for a given optimization configuration. This task can be interpreted as the determination of optimal values of p^* for a given problem design d or as an analysis of a regression meta-model [49].

3 Performance Measures to Compare Stochastic Optimizers

Before we can compare two algorithms, we have to specify a concrete measure for their comparison. Two main measurements play an important role in this context: effectivity and efficiency. Effectivity deals with the question whether the algorithm produces a desired effect. On the other hand, the measurement can be based on efficiency: does the algorithm produce the desired effects without waste?

There are many different measures for the goodness of an algorithm, i.e. the quality of the best solution, the percentage of runs terminated successfully, the number of iterations or time steps required to obtain the results, the robustness of the algorithm, or the distance between the best and the easiest found solutions. The performance measure under consideration should lead to a comparison well-defined, algorithmic, reproducible, and fair [50]. We list some typical measures to demonstrate that there is no canonical rule. If not mentioned explicitely, we will consider single criteria optimization problems defined as $\min\{f(x) : x \in \mathbb{D}\}$ with $f : \mathbb{D} \to \mathbb{R}$ and $\mathbb{D} \subseteq \mathbb{R}^D$.

- The mean best fitness can be defined as the average value of the best fitness values found at termination for one specific run configuration. Considering the quality of the best solution, it is a common practice to show a graph of the solution quality versus time.
- The first hitting time of an ϵ-environment of the global optimum of the objective function y^* can be defined as the expectation of the random time $T_\epsilon = \min\{t \geq 0 : B^{(t)} \leq y^* + \epsilon\}$ with $\epsilon \geq 0$, where $B^{(t)}$ is the best fitness function value known to the ES at time-step t [51]. As the best solution found during the complete run may not belong to the final population, this measure is of special interest when comma-strategies are analyzed.

- The hitting time of an ϵ-environment of y^* is also applicable to real-world optimization problems, where the exact optimum can only be approximated with a finite accuracy. Schwefel proposes the definition of an explicit border to determine successful runs or failures [52, 41].
- If the optimal solution is known, the percentage of run configurations terminated successfully (success rate) can be used to measure the performance of an algorithm [53]. This is related to an approach suggested by [41]: Test after each generation, if the interval of uncertainty of the variables has been reduced by at least 90 %: $|x_i^{(t)} - x_i^*| \leq \frac{1}{10}|x_i^{(0)} - x_i^*|, \quad \text{for } i = 1, \ldots, D$, where x^* denotes the optimum and the values x_i were initialized as

$$x_i^{(0)} = x_i^* + \frac{(-1)^i}{\sqrt{n}}, \qquad \text{for } i = 1, \ldots, D. \tag{4}$$

  Eiben and Jelasity (2002) discuss the relationship between success rate and mean best fitness. We will present a model based on the success rate in Sec. 4.6.
- Time-dependent measures such as the performance profile can be used to determine the computational effort [54].
- A measure to compute the quality-effort relationship can be defined as the ratio $r_{0.05} = t_{0.05}/t_{\text{best}}$, where $t_{0.05}$ denotes the time to produce a solution within 5 % of the best-found fitness function value, and t_{best} is the time to produce that best value [55].
- Robustness can be defined in many ways, i.e. as a good performance over a wide range of instances of one test problem or even over a wide range of different test problems.
- To measure the algorithm speed, the average number of evaluations to a solution can be used. The maximum number of evaluations can be used for runs finding no solutions.
- Schwefel (1988) defines the convergence velocity c as a progress measure of a single run as the logarithmic square root of the ratio of the best function value in the beginning $f^{(0)}$ and after t generations/iterations $f^{(t)}$: $c = \log(\sqrt{f^{(0)}/f^{(t)}})$. The normalized convergence velocity can be defined as $D/t \cdot c$ [20].
- Finally, we can annotate that [5] discuss performance measures for parallel algorithms.

Since runtimes depend on the computer system, measures for computational effort might be advantageous: Counting operations, especially for major subtasks such as fitness function calls can be mentioned in this context explicitly.

Schaffer et al. (1989) propose a technique to determine the total number of iterations N_{tot} and to prevent ceiling effects (a ceiling effect occurs when the fitness function values are nearly as good as possible): the number k belongs to the set $\mathcal{N}$, if at least 10% of the parameter design configurations p located the optimum y^* at least on average every second time after k iterations. The number of fitness function evaluations at which to compare different optimization algorithms is chosen as $N_{\text{tot}} = \min\{\mathcal{N}\} \mod 1000$.

We use the quality of the best solution found during an optimization run as a performance measure and propose two different approaches: the first approach (see Sec. 4.6) is based on the performance values found during the run, whereas the second approach interprets the number of successful runs as a random variable having a binomial distribution (see Sec. 4.6).

Since many optimization runs produce fitness function values that do not follow a Gaussian distribution, our conclusions will be based on generalized linear models (GLM) in this case. On the other hand, if the performance results are Gaussian, or can easily be transformed to Gaussian distributed values, classical analysis of variance (ANOVA) or regression models can be recommended.

4 Analysis: Design of Experiments

4.1 Experiments

Pre-experimental planning has a long tradition in other scientific disciplines. For instance, [57] present a checklist for the pre-experimental planning phases of an industrial experiment that covers the following topics: Objectives of the experiment, relevant background on response and control variables, a list of response variables and control variables, a list of factors to be "held constant", known interactions, proposed analysis techniques etc. The differences between analytical and empirical studies are discussed in [58]. A good empirical work must pass the following tests: *it must be both convincing and interesting* [58]. Moret (2002) gives a good characterization of "interesting": "always look beyond the obvious measures!" In this context, we recommend to include factors that should have no effect on the response such as the random seed r_0, see Eq. 5, in the model.

As we have classified important parameters of evolution strategies in Sec. 2, and have defined a measure for their comparison in Sec. 3, we can conduct experiments to assess the significance of single parameters such as population size or selective pressure. We should keep in mind that optimization runs are treated as experiments: we begin by formulating a hypothesis, then set up experiments to gather data that either verify or falsify this hypothesis. We will use guidelines from experimental algorithmics in our experimental studies [7]:

(G-1) Question: state a clear set of objectives. Formulate a question or a hypothesis. A typical question reads "Is the selective pressure $\nu = 5$ a good choice for the optimization problem under consideration?".
(G-2) Data collection: after an experimental design is selected, simply gather data. Do not modify the hypothesis until all data have been collected.
(G-3) Analysis: analyze the data to test the hypothesis stated above.
(G-4) Next Cycle: in the next cycle of experimentation a new hypothesis can be tested, i.e. "$\nu = 5$ is a good choice, because..."

This procedure is in accordance with Popper's position that "knowledge results when we accept statements describing experience that contradict and hence

refute our hypotheses; thus a deductive rather than an inductive relation holds between theoretical knowledge and experience. Experience teaches us by correcting our errors. Only hypotheses falsifiable by experience should count as scientific" [59].

4.2 How to Choose Exogenous Parameters

Johnson suggests to explain the corresponding adjustment process in detail, and therefore to include the time for the adjustment in all reported running-times to avoid a serious underestimate [50]. An important step to make this procedure more transparent and more objective is to use DOE techniques: they provide an algorithmical procedure to tune the exogenous parameter settings for the algorithms under consideration before the complex real-world optimization task is optimized or two algorithms are compared.

Besides the improved performance of an algorithm, fine-tuning of exogenous parameters might reveal information about its robustness. This may lead to new insights in the role of the offspring-parent ratio ν, or the relationship between recombination and mutation operator. Experimental design provides an excellent way of deciding which simulation runs should be performed so that the desired information can be obtained with the least amount of experiments [60, 61, 17, 49, 48].

4.3 Basic DOE Terminology

The input parameters and structural assumptions, that define a simulation model are called factors, the output value(s) are called response(s). The different values of parameters are called levels. Levels can be qualitative, i.e. selection scheme, or quantitative, i.e. population size. An experimental design is a set of factor level combinations. Kleijnen defines DOE as "the selection of combinations of factor levels that will be simulated in an experiment with the simulation model" [62]. One parameter design setting, cf. Eq. 1, is run for different pseudo-random number settings, resulting in replicated outputs. We will discuss linear regression models and their extensions, the so-called generalized linear models [63, 64].

4.4 Linear Regression Models

Generally, a simulation model can be represented as follows:

$$y = f_1(z_1, \ldots, z_k, r_0), \tag{5}$$

where f_1 is a mathematical function, e.g. $f_1 : \mathbb{R}^{k+1} \rightarrow \mathbb{R}$: given the values of the argument z_i and the random number seed r_0, the simulation program yields exactly one value. The Taylor series expansion yields the first order approximation $y = f_2 = \sum_{i=1}^{k} \beta_i z_i$. The last equation is the basis for regression models based on simulation data. Our goal is to use least square methods to estimate the linear model

$$y = X\beta + \epsilon, \tag{6}$$

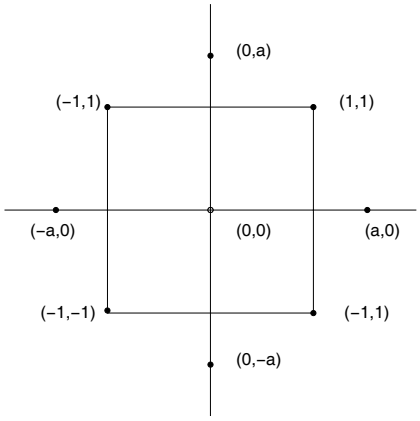

Fig. 2. Central composite design with axial runs. $a = \sqrt{2}$ gives a spherical CCD, that is: all factorial and axial design points are on the surface of a sphere of radius $\sqrt{2}$.

where y denotes a column vector with the n responses, ϵ is the vector of n error terms, and β denotes is the vector with q parameters β_j ($n \geq q$).[7] X is the $(n \times q)$ matrix of independent regression variables. x_0 is the dummy variable equal to **1**, the remaining $q - 1$ variables correspond to the simulation parameters z. Therefore, we obtain the $(n \times q)$ regression matrix

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1,q-1} \\ \vdots & & & & \vdots \\ 1 & x_{i1} & x_{i2} & \cdots & x_{i,q-1} \\ \vdots & & & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{n,q-1} \end{pmatrix}. \tag{7}$$

Experimental settings (designs), where the regression matrix X satisfies $XX' = nI$, are called orthogonal.[8] The least squares estimate of the regression coefficients in Eq. 6 is

$$\hat{\beta} = (X'X)^{-1}X'y. \tag{8}$$

Orthogonal designs simplify the computations. They lead to uncorrelated regression coefficients ($Cov(\beta_i, \beta_j) = 0$) and to a minimal variance of the predicted response in the region of interest. In the following, we use orthogonal designs with two factors for each level: 2^k factorial designs and 2^{k-p} fractional factorial designs, and orthogonal design where center points and axial points are added to the 2^k design: central composite designs (CCD) with axial runs (Fig. 2).

The commonly used one-factor-at-a-time method, where certain factors are varied one at a time, while the remaining factors are held constant, provides an

[7] The normality assumption (the error term ϵ has expectation $E(\epsilon) = 0$ and variance $V(\epsilon) = \sigma^2$) is discussed in Sec. 4.6.

[8] X' denotes the matrix transpose of X.

estimate of the influence of a single parameter at selected fixed conditions of the other parameters. Such an estimate may only have relevance under the assumption that the effect would be the same at other settings of the other parameters. This requires that effects of variables behave additively on the response over the ranges of current interest. Furthermore, interactions cannot be determined. Therefore, we do not recommend to use this method.

Factorial designs are more efficient than one-factor-at-a-time designs [17]. The variance stays unaltered in one-factor-at-a-time designs, whereas in factorial designs the variance decreases as the total number of experiments increases: we make use of each observation when we estimate an effect. If the influence of variables is not additive, factorial designs provide more precise results. Furthermore, unlike the one-factor-at-a-time method, factorial designs can detect and estimate interactions that measure the non-additiveness. Box et al. (1978) give an instructive example that explains the weakness of the classical one-factor-at a time design.
An important objection against 2^k designs is that non-linear effects remain undiscovered. But, this point does not apply in our case: we use 2^k designs to get an overview over the effects and their interactions, not to obtain the exact values. Furthermore, techniques to measure the goodness of the model fit can be applied.

A variable x is called standardized, if x ranges between -1 and $+1$. The original variables with range $[l, h]$ can be standardized using the linear transformation $x = a + bz$ with $a = (l + h)/2$ and $b = (l - h)/2$. In the remainder of this article z_i are the original or natural variables, whereas x_i are the standardized or coded variables.

Hence, the entry -1 in the regression matrix denotes a factor at its low level, and $+1$ a factor at its high level. The intuitive definition of a main effect of a factor A is the change in the response produced by the change in the level of A averaged over the levels of the other factors. The average difference between the effect of A at the high level of B and the effect of A at the low level of B is called the interaction effect AB of factor A and factor B.

4.5 Tuning

Generally, we suggest the following three-stage approach: screening – modeling – optimization. Before some examples are discussed in detail (see Sec. 5), the general outline is given.

1. Screening: Only the main effects but no interactions are considered. Therefore, we recommend fractional-factorial 2^{k-p} designs. These are orthogonal designs and require a moderate number of experiments. This means that the regression coefficients can be determined independently. If we cannot differentiate between two effects, these effects are called confounded. A 2^{k-p} design is of resolution R if no q-factor effect is confounded with another effect that has less than $R - q$ factors [60]. Roman numerals denote the corresponding design resolution. Our first experiments are based on resolution III designs (see Tab. 3). These designs ensure that no main effect is

confounded with any other main effect, but main effects can be confounded with two-factor interactions. These designs provide unbiased estimators of the regression coefficients of a first-order model and can easily be augmented to designs that enable the estimation of a second-order regression model, see also the next but one stage in our approach: optimization [18].

2. Modeling: Interactions are taken into account. At this stage, resolution IV or resolution V designs are recommended. A linear approximation (see Eq. 6) may be valid in a sub-domain of the full experimental area. In RSM, we determine the direction of improvement using the "path of the steepest descent" (minimization problem) based on the estimated first-order model [49]. The path of the steepest descent is perpendicular to the fitted first-order model. If no further improvement along the path of the steepest descent is possible, we can explore the area by fitting a local first-order model and obtain a new direction for the steepest descent. This step is repeated several times until we find the expected optimum area. There the linear model is inadequate and shows significant lack-of-fit. We cannot determine a direction of improved response in this case.
3. Optimization: Central composite designs and CCDs with additional axial runs, that require a relatively high number of runs, are often used at this experimental stage. They can be combined with response surface methods. We apply the standard techniques from regression analysis for meta-model validation [65]. A second-order model can be fitted in the expected optimum area. The optimal values are estimated by taking the derivatives of the second-order regression model. We combine in our approach DOE and RSM techniques, that are adapted to the special needs and restrictions of the optimization task.

As randomness is replaced by pseudo-randomness, we do not recommend to exclude outliers from the regression analysis. I.e., the decision whether the set of solutions $Y = \{y_1 = 3, y_2 = 3, y_3 = 3\}$ or $Z = \{z_1 = 0, z_2 = 3, z_3 = 6\}$ is better depends on the optimization task. Removing the potential outliers z_1 or z_3 from Z may destroy valuable information.

4.6 Generalized Linear Models

Linear models, regression analysis, and analysis of variance as discussed so far are applicable to problems having errors that are Gaussian. In many situations the optimization practitioner has to face response values that follow some skewed distribution or have non-constant variance. To deal with non-normal responses, data transformations are often recommended, although the choice of an adequate transformation can be difficult. Draper and Smith (1998) discuss the need for transformation and present different transformation methods. Since the transformation may result in incorrect values for the response value, i.e. $\log Y$, if $Y < 0$, generalized linear models provide an alternative [64].

The linear model, see Eq. 6, with independent normal Y_i variables that have constant variance σ^2 and expectation $E(Y) = \mu$, where $\mu = X\beta$, is a special

case of the generalized linear model (GLM) with the following components [64]:
– Response variables Y_i that are assumed to have a distribution from the exponential family, and
– a monotone link function g that determines the relationship between the mean μ and the linear predictor: $g(\mu) = X\beta$.
The GLM regression model is given by $E(Y) = \mu = g^{-1}(X\beta)$.

A Model Based on the Quality of the Best Solution We will give some hints on how to perform the regression analysis based on GLMs: Histograms can give first indications, whether or not it might be adequate to use a specific distribution (i.e. a Gamma distribution might be better suited than the Gaussian distribution). Next we can consider quantile-quantile-plots (QQ-plots) or a Kolmogorov–Smirnoff test (KS test) to support the assumption. Stepwise model selection, by adding or removing terms, can be used to find a more parsimonious regression model [63] and to estimate the regression coefficients. The statistical software package `R` provides functions such as `stepAIC` that automate this selection process. Functions such as `qq.plot` can be used to plot the data against any reference distribution [66].

A Logistic Regression Model Based on the Success Rate Whether or not the optimization run has located a pre-specified optimum can be used as a performance measure for algorithms. In this case, where the outcome variable can take only two values, a linear regression model is not appropriate, but a logistic regression model might be adequate. Let $\mathbf{D}$ denote the set of all parameter designs and $\mathbf{P}$ the set of all problem configurations. Each (optimization run, problem)-combination generates a binary response value: $\mathbf{D} \times \mathbf{P} \to \{0, 1\}$, with $d \in \mathbf{D}$ and $p \in \mathbf{P}$ as defined in Eq. 1 and Eq. 3 respectively. As N_{exp} optimization runs for each factor-level setting will be performed, the number of successful runs can be seen as a random variable having a binomial distribution.

For an introduction into logistic regression the reader is referred to [67]. Myers and Hancock (2001) present an example that uses a genetic algorithm to solve consistent labeling problems. The logistic regression model might also be applicable if the outcome variable is disturbed by noise as in the elevator simulation model (this model will be discussed later).

4.7 Tree Based Methods

Breiman et al. (1994) introduce regression trees as a "flexible non-parametric tool to the data analyst's arsenal." They are used for screening variables and for checking the adequacy of regression models [69]. The construction of regression trees can be seen as a type of variable selection [63]. Consider a set of predictor variables X and a quantitative response variable Y. A regression tree is a collection of rules such as "if $x_1 \leq 5$ and $x_4 \in \{A, C\}$, then the predicted value of Y is 14.2", that are arranged in a form of a binary tree (see, e.g., Fig. 3). Recursively splitting the data in each node builds up a binary tree. The partitioning

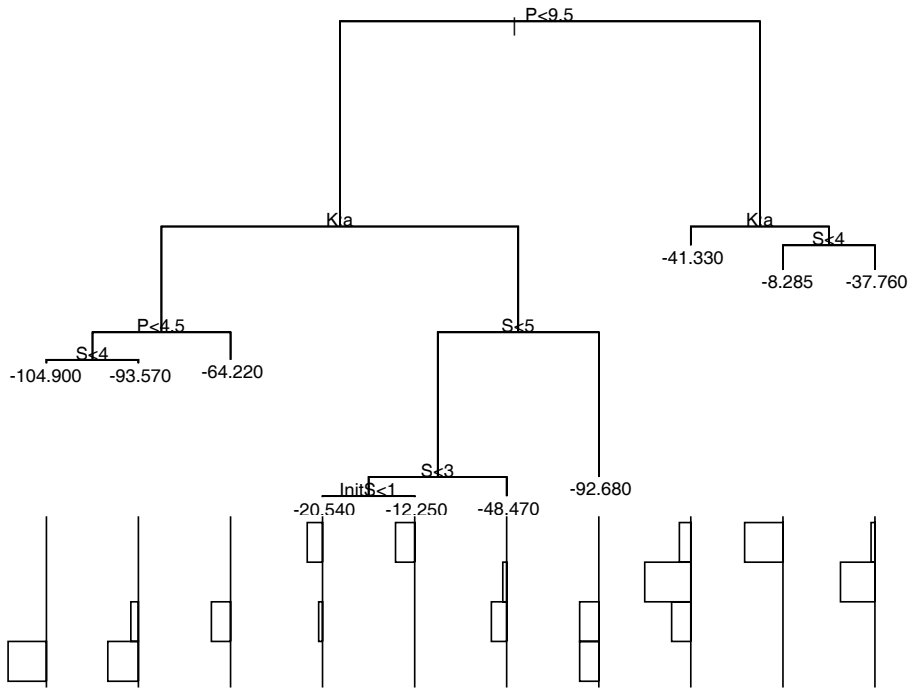

Fig. 3. Experiment ES4. Tile tree plot visualizing tree regression. Evolution strategy optimizing the 12 dimensional sphere function. P stands for the population size μ, K denotes the selection mechanism κ, S the selective pressure μ, and $InitS$ stands for the initial step size $\sigma^{(0)}$. Further symbol names are explained in Tab. 2. The left subtree of a node contains the configurations that fulfill the condition in the node. It is easy to see that smaller populations improve the algorithm's performance. The corresponding experiment is described in Sec. 5.1. A nonuniform spacing was chosen: more important node splits result in larger spaces between nodes.

algorithm stops when the node is homogeneous or the node contains too few observations. Compared to linear models, tree-based models are easier to interpret when qualitative and quantitative predictors are in the model. The endpoint for a tree is a partition of the space of possible observations. The `S-Plus` function `tile.tree` is used to display class probabilities across the leaves of a tree.

5 Examples

In the following examples we will investigate minimization problems $f : \mathbb{D} \rightarrow \mathbb{R}, \quad \mathbb{D} \subseteq \mathbb{R}^D$. The region of interest is defined as

$$I := [a_1, b_1] \times \ldots \times [a_D, b_D] \subseteq \mathbb{D}, \tag{9}$$

with the center point $z_i = (a_i + b_i)/2, \quad i, \ldots, D$, as depicted in Fig. 2. The optimization response is approximated in the region of interest by the first-order regression model $y = X\alpha + \epsilon$, or by the coded model $y = X\beta + \epsilon$, cf. Eq.6.

Imagine a situation in which an optimization practitioner can only perform a few preliminary experiments to find a suitable ES parameter setting. Since the ES discussed in Sec. 2 has 9 different exogenous parameters, a full factorial 2^k design would require 512 optimization runs. Thus, the experimenter decides to set up a 2^{9-5}_{III} fractional factorial design, that requires only 16 optimization runs. The first $4 = 9 - 5$ columns in the design matrix are identical to the columns of a full factorial 2^4 design matrix. The remaining four columns are constructed by "multiplications" that are based on the defining relations [61]: $E = ABC$, $F = BCD$, $G = ACD$, $H = ABD$, and $J = ABCD$. The resulting design is shown in Tab. 3. Box et al. (1978) give rules for constructing fractional factorial designs.

5.1 Tuning: Evolution Strategy and Sphere Function

The 12 dimensional sphere function $f : \mathbb{R}^D \rightarrow \mathbb{R}$, $y = f(x) = \sum_{i=1}^{D} x_i^2$, has been chosen to demonstrate the experimental analysis methodology.[9]
(G-1) Question: *is the standard ES parameterization a reasonable choice to optimize the sphere function?*
(G-2) Data collection: based on the standard ES configuration from Tab. 2, the 2^{9-5}_{III} fractional factorial design from Table 5 was used (Experiment ES1). The

[9] As $D = 12$ was used in other contexts, see e.g. [70], this setting was chosen to improve the comparability.

regression matrix reads:

$$X = \begin{pmatrix}
1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 \\
1 & 1 & 1 & -1 & -1 & -1 & 1 & 1 & -1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 \\
1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 \\
1 & -1 & 1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & 1 \\
1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 \\
1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 \\
1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 \\
1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & -1 \\
1 & -1 & 1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}. \tag{10}$$

The interval $[-1\text{Level}, +1\text{Level}]$ from Tab. 4 contains the recommended value from Tab. 2: i.e., we have chosen the region of interest $I_1 = [10, 20]$ for μ, as the recommended value $\mu = 15$ lies in the interval I_1. Performing a stepwise model selection by AIC gives the first-order model with coefficients[10] for the coded variables in Tab. 4

$$\log(y) = -46.38 + 0.7795x_1 + 1.222x_2 + 1.555x_3 + 0.846x_4 - 1.909x_7 - 1.05x_9. \tag{11}$$

Regression analysis reveals that x_1, x_2, x_3, x_4, and x_7 are significant at the 99% level.[11]

Before we start the search along the path of the steepest descent, the adequacy of the regression model is checked, and a check for interactions is performed. Interactions between the factors result in lack-of-fit, that can be detected by analysis of variance (ANOVA) methods. For instance, the performance of plus strategies might be improved by increasing the population size on the one hand, whereas comma strategies require a higher selection pressure on the other hand. If this occurs, both strategies will be treated separately.

Starting from the center point $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 0$ in the coded variables or $Z_1 = \big(\mu = 15, \nu = 7, \sigma^{(0)} = 3.0, n_\sigma = 1, c_\tau = 1, \rho = m, r_x = d, r_\sigma = i, \kappa = 1\big)$ in the original variables, we perform a line search (Experiment ES2) in the direction of the steepest descent that is given by $-(\hat{\beta}_1, \ldots, \hat{\beta}_D)$. To determine the step-sizes Δx_i, we select the variable x_j that has the largest absolute regression coefficient: $j = \arg\max_i |\hat{\beta}_i|$. The increment in the other variables is $\Delta x_i = -\hat{\beta}_i / (|\hat{\beta}_j| / \Delta x_j)$, $i = 1, 2, \ldots, k$; $i \neq j$. The model includes five quantitative variables $(\mu, \nu, \sigma^{(0)}, c_\tau)$ and four qualitative variables $(n_\sigma, r_x, r_\sigma, \kappa)$.

[10] μ and ν are treated as quantitative factors, their values are rounded to the nearest whole number to get a set of working parameters.

[11] If not mentioned explicitly, we shall refer to significance as significance at the 99% level.

Table 3. Fractional factorial 2^{9-5}_{III} design. This design is used for screening the ES parameters. Concrete values are shown in Tab. 5

	A	B	C	D	E=ABC	F=BCD	G=ACD	H=ABD	J=ABCD
1	−	−	−	−	−	−	−	−	+
2	+	−	−	−	+	−	+	+	−
3	−	+	−	−	+	+	−	+	−
4	+	+	−	−	−	+	+	−	+
5	−	−	+	−	+	+	+	−	−
6	+	−	+	−	−	+	−	+	+
7	−	+	+	−	−	−	+	+	+
8	+	+	+	−	+	−	−	−	−
9	−	−	−	+	−	+	+	+	−
10	+	−	−	+	+	+	−	−	+
11	−	+	−	+	+	−	+	−	+
12	+	+	−	+	−	−	−	+	−
13	−	−	+	+	+	−	−	+	+
14	+	−	+	+	−	−	+	−	−
15	−	+	+	+	−	+	−	−	−
16	+	+	+	+	+	+	+	+	+

Table 4. Evolution strategy: symbols and levels. Values chosen with respect to the default settings from Tab. 2.

Symbol	Factor	Standardized	−1 Level	+1 Level	Type
μ	P	x_1	10	20	quantitative
ν	S	x_2	5	10	quantitative
$\sigma^{(0)}$	$InitS$	x_3	1	5	quantitative
n_σ	$NSigma$	x_4	1	12	qualitative
c_τ	$TauMult$	x_5	1	2	quantitative
ρ	Rho	x_6	b	m	qualitative
r_x	$XReco$	x_7	i	d	qualitative
r_σ	$SReco$	x_8	i	d	qualitative
κ	K	x_9	−1	1	qualitative

Table 5. Fractional factorial design for evolution strategies.

	μ	ν	$\sigma^{(0)}$	n_σ	c_τ	ρ	r_x	r_σ	κ
1	10	5	1	1	1	2	i	i	1
2	20	5	1	1	2	2	d	d	-1
3	10	10	1	1	2	10	i	d	-1
4	20	10	1	1	1	20	d	i	1
5	10	5	5	1	2	10	d	i	-1
6	20	5	5	1	1	20	i	d	1
7	10	10	5	1	1	2	d	d	1
8	20	10	5	1	2	2	i	i	-1
9	10	5	1	12	1	10	d	d	-1
10	20	5	1	12	2	20	i	i	1
11	10	10	1	12	2	2	d	i	1
12	20	10	1	12	1	2	i	d	-1
13	10	5	5	12	2	2	d	d	1
14	20	5	5	12	1	2	i	i	-1
15	10	10	5	12	1	10	d	i	-1
16	20	10	5	12	2	20	i	d	1

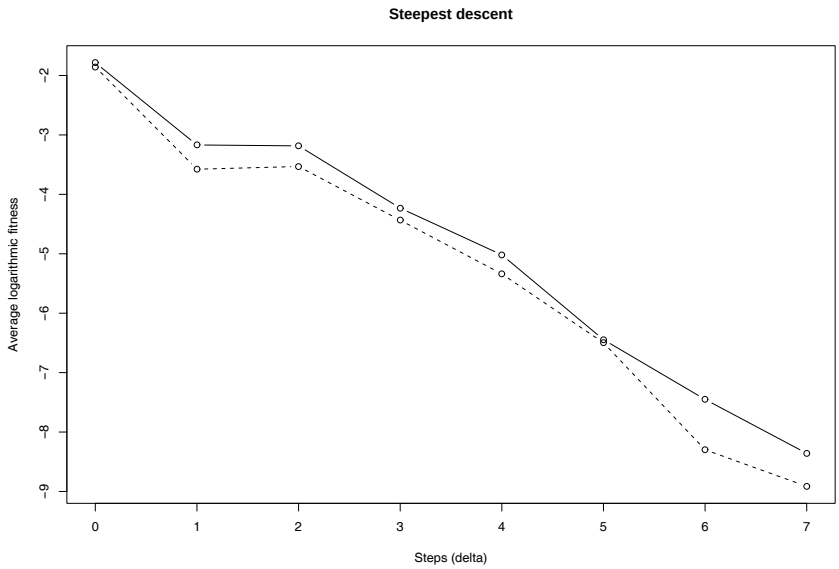

Fig. 4. Experiment ES2. Steepest descent (plus selection). 12 dimensional sphere. MaxIter = 5,000. 5 repeats. Stopping rule: $\sigma^{(0)} < 0$. Lines are shown to enhance readability. The dotted line shows the median, the solid line the mean of the response from 5 repeat runs. The comma selection variant exhibits a similar behavior.

Table 6. Experiment ES2. Steepest descent (plus selection). 12 dimensional sphere function. See also Fig. 4.

	$\sigma^{(0)}$	ν	μ	$\sigma^{(0)}$	ν	μ	Mean	Median
Steps	Coded	Coded	Coded	Original	Original	Original	Response	Response
Δ	x_1	x_2	x_3	z_1	z_2	z_3	$\log(y)$	$\log(y)$
	0	0	0	3.0	8	15	-1.784	-1.856
Δ	-0.2	-0.15	-0.1	2.6	7	14	-3.169	-3.577
2Δ	-0.4	-0.3	-0.2	2.2	7	14	-3.184	-3.531
3Δ	-0.6	-0.45	-0.3	1.8	6	14	-4.231	-4.435
4Δ	-0.8	-0.6	-0.4	1.4	6	13	-5.018	-5.339
5Δ	-1.0	-0.75	-0.5	1.0	6	12	-6.445	-6.497
6Δ	-1.2	-0.9	-0.6	0.6	5	12	-7.451	-8.298
7Δ	-1.35	-1.05	-0.7	0.2	5	12	-8.359	-8.915

For qualitative factors with significant effects the "better" levels were chosen. The values of qualitative factors with small effects on the response were chosen rather subjectively. Tab. 6 shows the numerical values, whereas Fig. 4 displays the change in the response during the line search. Before the initial sigma value $\sigma^{(0)}$ becomes negative, the search is stopped. The improved (original) parameter vector reads:

$$Z_2 = \left(\mu = 12, \nu = 5, \sigma^{(0)} = 0.2, n_\sigma = 1, c_\tau = 1, \rho = m, r_x = d, r_\sigma = i, \kappa = 1\right).$$

A 2^4 full factorial design $(\mu, \nu, \sigma^{(0)}, \kappa)$ that was chosen next (Experiment ES3), reveals that only the factors μ and ν have significant effects on the response. As μ and ν are important factors, a central composite design with axial runs is used in the following step (Experiment ES4). The CCD combines a 2^k factorial design with n_F runs, n_C center runs, and $2k$ axial runs. The distance a of the axial run points from the design center was set to $\sqrt{2}$. As all the factorial and axial design points are on the surface of a sphere of radius $\sqrt{2}$, this design is called a spherical CCD, see Fig. 2.

The corresponding regression tree model is shown in Fig. 3 and can be read as follows: To predict the fitness from a given parameter configuration, one follows the path from the top node (root) of the tree to a leaf. For instance, starting from the top node and following the left branches (population size μ less than 9.5, plus selection: κ, population size μ less than 4.5, and selective pressure ν less than 4) the terminal node with a predicted fitness function value of $log(y) = -104.9$ is reached.

The plus selection scheme is advantageous in this case. The parameters population size μ, selective strength ν, and initial standard deviations $\sigma^{(0)}$ are the subject of the further analysis. We can conclude from the regression analysis that was based on a 2^3 central composite design (Experiment ES5) that a further decrease of the population size and of the selective pressure might be beneficial:

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -141.4920     2.6745  -52.90   <2e-16 ***
P              5.4705     0.1881   29.08   <2e-16 ***
S              6.9551     0.4703   14.79   <2e-16 ***
InitS          2.0380     1.0452    1.95    0.058 .
```

The initial step-size $\sigma^{(0)}$ has no significant effect any more. This result corresponds with the conclusions that might be drawn from the tree based regression analysis. Figure 5 shows a design plot of the median of the response y at each of the levels of the factors in the first experiment. The horizontal line indicates the treatment median of the data. It is obvious that the factors μ and ν have a strong effect on the response.

Instead of performing a second line search, we use a CCD with axial runs to take a closer look at the fitness landscape (Experiment ES6). Each configuration was repeated 10 times. The resulting graph is shown in Fig. 6. A comparison of these results to the results from the same design with 50 repeats showed no significant difference.

The final ES parameterization reads:

$$Z_3 = \left(\mu = 3, \nu = 1, \sigma^{(0)} = 0.1, n_\sigma = 1, c_\tau = 1, \rho = m, r_x = i, r_\sigma = i, \kappa = -1\right).$$

(G-3) Analysis: a numerical comparison of the first

```
   Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.007042 0.037910 0.049710 0.068540 0.092020 0.197100
```

and the improved function values

```
   Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
5.324e-64 1.155e-59 1.252e-57 6.472e-51 7.544e-56 5.165e-49
```

reveals a significant improvement. The left boxplot in Fig. 7 shows a graphical comparison of the first and the improved design. As we have found a better parameterization, we will answer Question (G-1) in the negative. Thus, a (3+3)-ES with one step-size, intermediate recombination of object and strategy parameters works better than the "standard" ES presented in [12]. This result is not surprising, since this standard was chosen as a "good" parameterization on average for many problems and not especially for the sphere model.
(G-4) Next cycle: as function values become rather small, numerical characteristics of the machine the program is running on (i.e. the machine's precision), and the ceiling effect might influence our analysis. Therefore, we examine more complex optimization problems in the following.

The result found so far does not justify the conclusion that Z_3 is the optimal design. The intention of this paper is to give the optimization practitioner a framework on how to set up algorithms with working parameter configurations. Further optimization of Z_3 is possible, but this is beyond the intention of this paper.

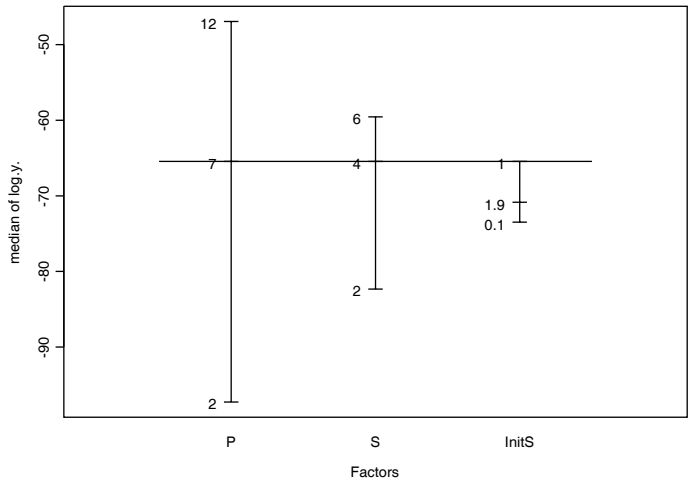

Fig. 5. Experiment ES5. Design plot. 12 dimensional sphere. Plus strategy. MaxIter = 5,000. 5 repeats. The regression tree analysis (not shown here) reveals a similar result.

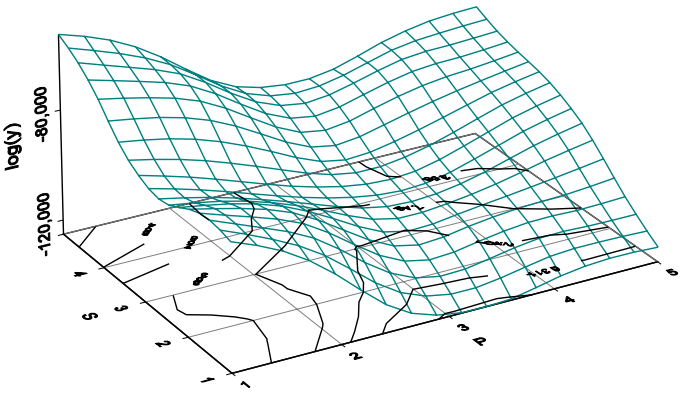

Fig. 6. Experiment ES6. Spline plot. 12 dimensional sphere. MaxIter = 5,000. 10 repeats. CCD with axial runs. 12 dimensional sphere. MaxIter = 5,000. 10 repeats.

5.2 Tuning: Evolution Strategies and Rosenbrock's Function

The 2-dimensional "banana" function after [71] $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ will be the subject of the next analysis.
(G-1) Question: *is the standard ES parameterization a reasonable choice to optimize Rosenbrock's function?*
(G-2) Data collection: the 2^{9-5}_{III} fractional factorial design (Tab. 5) reveals that the plus strategy outperforms the comma strategy for this specific optimization run configuration. As we can neglect the corresponding factor κ, a 2^{8-4}_{IV} fractional factorial design was used to analyze the remaining eight factors (Experiment ES7).

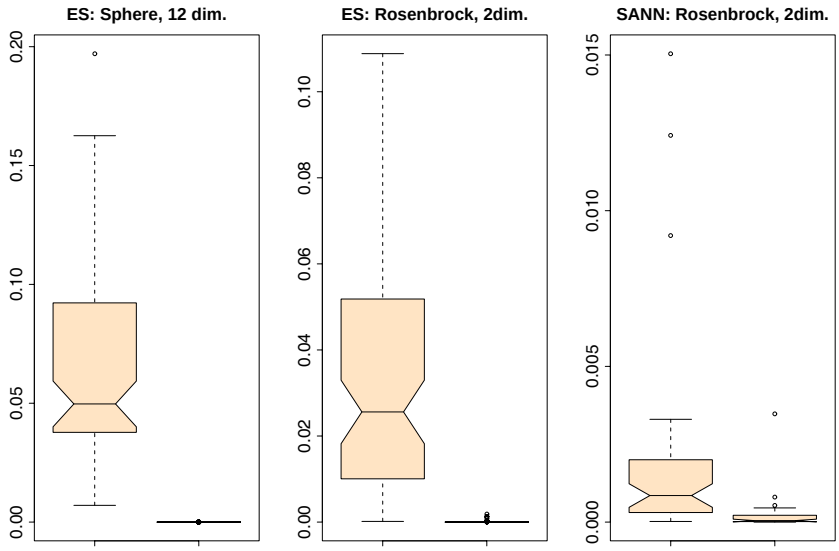

Fig. 7. Box plot. Comparison of the first (left) and improved (right) configurations. ES: 12 dimensional sphere, Rosenbrock 2 dimensional. SANN: Rosenbrock 2 dimensional. MaxIter = 5,000. 80 repeats.

Tree based regression and further analysis lead to the improved parameter setting

$$Z_1 = \left(\mu = 4, \nu = 9, \sigma^{(0)} = 1.0, n_\sigma = 1, c_\tau = 1, \rho = m, r_x = i, r_\sigma = i, \kappa = -1\right).$$

The response surface generated by a 2^2 full factorial design with center points (Experiment ES8) leads to the improved ES parameterization:

$$Z_2 = (\mu = 5, \nu = 10, \sigma^{(0)} = 0.37, n_\sigma = 1, c_\tau = 1, \rho = b, r_x = d, r_\sigma = i, \kappa = -1). \tag{12}$$

The boxplot in the center of Fig. 7 compares the fitness distributions of the initial and the improved algorithm.
(G-3) Analysis: since we have found a parameterization for an ES that performs significantly better than the ES with the standard parameterization, we have to negate (G-1). (G-4) Next cycle: as we have obtained a "good" solution (cf. Eq. 12), we decided to stop the parameter optimization process here.

Table 7. Simulated annealing. Full factorial 2^3 design

	A	B	C	tmax	temp	parscale
1	−	−	−	5	5	1
2	+	−	−	10	5	1
3	−	+	−	5	10	1
4	+	+	−	10	10	1
5	−	−	+	5	5	2
6	+	−	+	10	5	2
7	−	+	+	5	10	2
8	+	+	+	10	10	2

5.3 Comparison: Evolution Strategy and Simulated Annealing

The method SANN is taken from the freely available software package `R` [66]. It is by default a variant of simulated annealing given in Belisle (1992). Simulated-annealing belongs to the class of stochastic global optimization methods and uses only function values. It also works for non-differentiable functions. This implementation uses the Metropolis function for the acceptance probability.The next candidate point is generated from a Gaussian Markov kernel with scale proportional to the actual temperature. Temperatures are decreased according to the logarithmic cooling schedule as given in [11]. The three exogenous parameters for the standard simulated annealing are shown in Tab. 8.
(G-1) Question: *do the default values from Tab. 8 provide a good parameterization for the SANN algorithm to optimize Rosenbrock's function?*
(G-2) Data collection: starting from the default configuration

$$p_{\text{SANN}} = (t = 10, m = 10, s = 1)\,,$$

the improved parameter vector reads: $p_{\text{SANN}^*} = (t = 1, m = 10, s = 0.2)$. The response from the improved SANN reads (Experiment SANN1):

```
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
1.242e-07 1.736e-05 4.871e-05 1.943e-04 1.882e-04 3.476e-03.
```

(G-3) Analysis: since we have found a parameterization that performs significantly better than the SANN with the standard parameterization from Tab. 8, we have to negate (G-1).

(G-4) Next cycle: as we have obtained a "good" solution, we decided to stop the parameter optimization process here.

Table 8. Exogenous parameters for the standard simulated annealing (SANN). Description taken from the `R` documentation [66].

Symbol	Factor	Parameter	Default Values
t	temp	Controls the SANN method. Starting temperature for the cooling schedule	10
m	tmax	Number of function evaluations at each temperature for the SANN method	10
s	parscale	A vector of scaling values for the parameters. Optimization is performed on par/parscale and these should be comparable in the sense that a unit change in any element produces about a unit change in the scaled value.	$(1, 1, \ldots, 1) \in \mathbf{R}^D$

Now that we have tuned the evolution strategy and the simulated annealing, we are able to compare their performances.
(G-1) Question: *do the tuned ES and the tuned SANN reveal a similar performance while optimizing Rosenbrock's function?*
(G-2) Data collection: no further data have to be collected, the comparison can be based on the data generated in the previous experiments.
(G-3) Analysis: A graphical comparison of the fitness values obtained from the ES

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
8.941e-18 2.350e-11 5.565e-09 8.310e-05 6.084e-06 1.873e-03
```

and the SANN algorithm with improved parameter settings for the 2 dimensional Rosenbrock function is shown in Fig. 8. Boxplots (A) and (B) in Fig. 9 show a comparison, that might lead to the conclusion that evolution strategies attain better results than SANN while optimizing Rosenbrock's function.
(G-4) Next cycle: keeping in mind that this comparison is only valid for the experimental settings mentioned above, it might be interesting to extend these investigations for a better understanding.

5.4 Tuning: Evolution Strategy and Elevator Group Control

In the previous sections, we have applied DOE methods to artificial test functions. The following example illustrates how these methods can be applied to real-world optimization problems. We have chosen the elevator supervisory group (ESGC) problem to demonstrate the applicability of the experimental analysis approach to real-world optimization problems. The ESGC problem subsumes the following problem: how to *assign elevators to passengers* in real-time while optimizing different elevator configurations with respect to overall service quality,

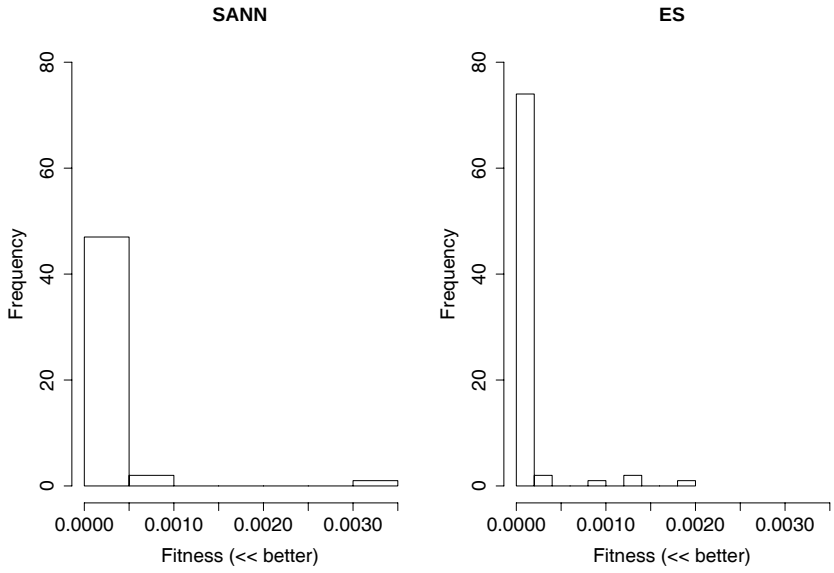

Fig. 8. Comparison of the improved parameter settings. 2 dimensional Rosenbrock function. MaxIter = 5,000. 50 repeats.

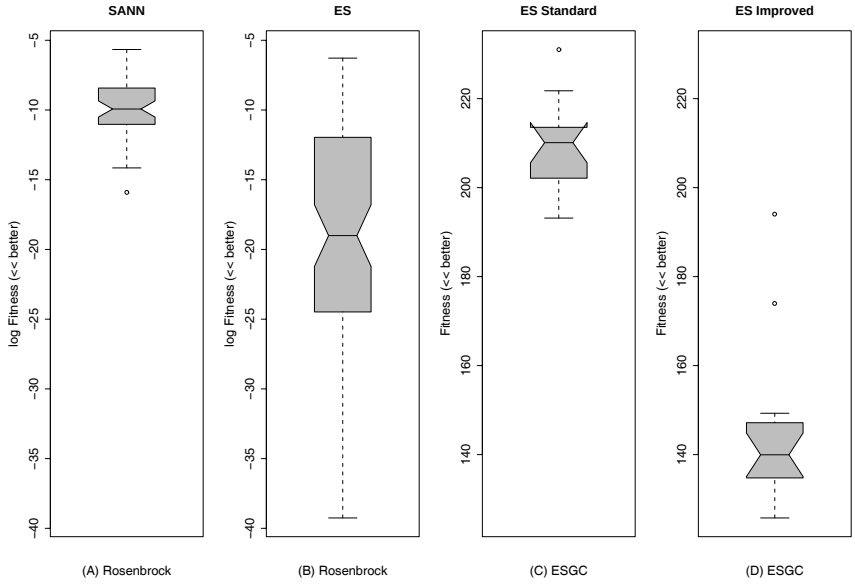

Fig. 9. Comparison of the improved ES (A) and SANN (B) parameter settings. 2 dimensional Rosenbrock function. Standard ES (C) and tuned ES (D) optimizing the ESGC problem.

traffic throughput, energy consumption etc. One main goal in designing a better controller is to minimize the time passengers have to wait until they can enter an elevator car after having requested service. This time-span is called the waiting time. The following simulations are based on a controller that was developed by Fujitec, one of the world's leading elevator manufacturers. It is trained by use of a set of neuro-fuzzy controllers. The neural network (NN) structure and the neural weights determine a concrete control strategy for different traffic situations [72, 27]. The network structure as well as many of the weights remain fixed, only some of the weights on the output layer can be modified and optimized. A discrete-event based elevator group simulator permits computing the controller's performance. The identification of globally optimal NN weights is a highly complex task since the objective function topology is highly non-linear and highly multimodal. It is stochastically disturbed due to the nondeterminism of service calls, and dynamically changing with respect to traffic loads. Gradient based optimization techniques cannot be applied successfully to this optimization problem. The computational effort for single simulator runs limits the maximum number of fitness function evaluations to the order of magnitude 10^4. The problem parameters read in vector notation: $d_{\mathrm{ESGC}} = \left(D = 36, x_i^{(0)} = (-1)^i/\sqrt{D}, N_{\mathrm{tot}} = 1000\right)$.
(G-1) Question: *is the standard ES parameterization a reasonable choice to optimize the ESGC problem?*
(G-2) Data collection: the first step in our analysis is based on the generic 2^{9-5}_{III} fractional factorial design (Tab. 3 and Tab. 5). Screening reveals that $\sigma^{(0)}$,n_σ, c_τ, r_x, and κ are important factors (Experiment ES9). As the function values are disturbed by noise, a logistic regression with a binary response Z (success/failure as described in Sec. 3) was chosen [53]: $Z = 1$, if $Y < 200$ and $Z = 0$ otherwise. Based on regression tree analysis and on a logistic regression model, we obtained the improved parameter vector:

$$Z_1 = \left(\mu = 2, \nu = 5, \sigma^{(0)} = 5, n_\sigma = 36, c_\tau = 2, \rho = b, r_x = d, r_\sigma = i, \kappa = 1\right).$$

(G-3) Analysis: The standard parameterization from Tab. 2 leads to the following response values (16 repeats)

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
193.2   202.3   210.1   209.3   213.5   231.1,
```

whereas the improved algorithm was able to find the following results:

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
125.8   134.9   140.0   144.6   146.5   194.0.
```

The boxplots on the right in Fig. 9 compare the performance of the standard ES configuration to the performance of the tuned ES. As we were able to enhance the algorithm's performance significantly after only a small number of simulation runs, we have to negate (G-1).
(G-4) This example illustrates, how to improve algorithms' performance for complex real-world optimization problems. Advanced investigations might lead to further improvements and are the subject of current research.

6 Discussion of the Method

The method proposed in this paper may fail for many reasons, e.g. if a problem is too hard for an algorithm or if the algorithm is very sensitive to parameter changes. However, any methodology to determine a good parameter setting will fail in this case.

The optimization practitioner is interested in robust algorithms that show a good performance over a wide range of parameter settings. Although standard parameterizations as shown in Tab. 2 can give valuable hints, difficulties may arise in the selection of the area of interest I, cf. Eq. 9. The first order model that is used to approximate the response surface function locally, may be inadequate. This can be solved by decreasing the size of the region of interest, by approximating the response function by a second-order model, or by increasing the number of repeats.

The choice of a stopping rule for the line search is not trivial. The simplest rule ends the line search when no further improvement is observed. This rule is not optimal under noise that occurs in many real-world optimization problems [73]. Fortunately, the problems mentioned here are well-known optimization problems and the subject of current research.

Carrying our approach to an extreme, one might conclude that every optimization problem requires a specific algorithm. This is in fact true, but developing an algorithm maybe very time consuming, so that the practitioner may prefer the tuning of an already existing algorithm. Anyhow, even the new algorithm can be tuned with the methodology presented in this paper.

7 Summary and Outlook

This paper introduces a framework that can be used to improve the acceptance and quality of research in the field of evolutionary computation. It summarizes state of the art techniques for EA parameter tuning, that can be beneficial in the following situations:

- Real-world optimization problems permit only a few preliminary experiments to find good EA parameter settings. Since the commonly used "one factor at a time approach" is rather inefficient and ineffective, we recommend DOE methods.
- As demonstrated in Sec. 5, there exist no "standard" parameterization for EA and related algorithms. Each algorithm should be tuned a priori to enable a fair and more objective comparison of different optimization algorithms.

Referring to the methodology proposed by [29], GLM are used to handle responses that are not Gaussian. If the responses are normally distributed, classical regression analysis can be applied [17]. Henceforth, a regression tree approach was proposed as a valuable tool to detect significant effects.

The experimental results from Sec. 5 lead to new questions and conjectures such as: "The ESGC problem requires relatively large initial step-sizes in contrast to the sphere function and to Rosenbrock's function. Is this behavior caused

by many local optima?" Hence, combining experimental data with exploratory techniques to visualize and analyze structure helps to set up empirical studies that provide answers to relevant research questions and might lead to new theories. A similiar approach is used in the highly recommendable book "Stat Labs - Mathematical Statistics Through Applications" by Nolan and Speed (2000). The case studies presented in their book should raise interesting scientific questions. Figuring out how to answer a question is the starting point for developing a theory.

The "experimental analysis of search heuristics" methodology presented here is related to other empirical approaches that are based on experiment design and explorative data analysis. It provides a good starting point for further analysis of search heuristics and is a valuable framework for understanding and improving these algorithms.

Acknowledgments

T. Bartz–Beielstein's research was supported by the DFG as a part of the collaborative research center "Computational Intelligence" (531).

References

1. P. Bentley. ISGEC workshop on standards at GECCO 2002, 2002. http://www.cs.ucl.ac.uk/staff/P.Bentley/standards.html.
2. A.E. Eiben and M. Jelasity. A critical note on experimental research methodology in EC. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC'2002)*, pages 582–587. IEEE Press, 2002.
3. P. Cohen. A survey of the eighth national conference on artificial intelligence: Pulling together or pulling apart? *AI Magazine*, 12(1):16–41, 1991.
4. R. A. Fisher. *The Design of Experiments*. Oliver and Boyd, Edinburgh, 1935.
5. R. Barr and B. Hickman. Reporting computational experiments with parallel algorithms: Issues, measures, and experts' opinions. *ORSA Journal on Computing*, 5(1), 1993.
6. J. Hooker. Testing heuristics: We have it all wrong. *Journal of Heuristics*, 1(1):33–42, 1996.
7. B. M. E. Moret. Towards a discipline of experimental algorithmics. In M.H. Goldwasser, D.S. Johnson, and C.C. McGeoch, editors, *Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges*, DIMACS Monographs 59, pages 197–213, Providence, 2002. American Mathematical Society.
8. W.D. Kelton. Experimental design for simulation. In J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick, editors, *Proceedings of the 2000 Winter Simulation Conference*, 2000.
9. J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435, 1989.
10. H.-G. Beyer and H.-P. Schwefel. Evolution strategies – A comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

11. C. J. P. Belisle. Convergence theorems for a class of simulated annealing algorithms. *Journal Applied Probability*, 29:885–895, 1992.
12. T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, 1996.
13. T. Beielstein. Tuning evolutionary algorithms. Technical Report 148/03, Universität Dortmund, 2003.
14. P. Cohen, I. P. Gent, and T. Walsh. Tutorial given at AAAI, ECAI and Tableaux conferences in 2000, 2000.
15. D.E. Gregory, L.-X. Gao, A.L. Rosenberg, and P.R. Cohen. An empirical study of dynamic scheduling on rings of processors. In *8th IEEE Symp. on Parallel and Distr. Processing*, pages 470–473, 1996.
16. D. C. Montgomery. *Design and Analysis of Experiments*. John Wiley & Sons, New York, NY, 5th edition, 2001.
17. J. Kleijnen. *Statistical Tools for Simulation Practitioners*. Marcel Dekker, New York, 1987.
18. J. P. C. Kleijnen. Experimental design for sensitivity analysis, optimization, and validation of simulation models. In J. Banks, editor, *Handbook of simulation*. Wiley, New York, 1997.
19. J. P. C. Kleijnen and O. Pala. Maximizing the simulation output: a competition. *Simulation*, 73:168–173, September 1999.
20. F. Kursawe. *Grundlegende empirische Untersuchungen der Parameter von Evolutionsstrategien — Metastrategien*. Dissertation, Fachbereich Informatik, Universität Dortmund, 1999.
21. J. D. Schaffer, R. A. Caruana, L. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, CA, 1989. Morgan Kaufman.
22. A.E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Trans. on Evolutionary Computation*, 3(2):124–141, 1999.
23. D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.
24. D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon. Optimization by simulated annealing: an experimental evaluation. Part i, graph partitioning. *Operations Research*, 37(6):865–892, 1989.
25. D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon. Optimization by simulated annealing: an experimental evaluation. Part ii, graph coloring and number partitioning. *Operations Research*, 39(3):378–406, 1991.
26. C. Demetrescu and G. F. Italiano. What do we learn from experimental algorithmics? In *Mathematical Foundations of Computer Science*, pages 36–51, 2000.
27. T. Beielstein, C.P. Ewald, and S. Markon. Optimal elevator group control by evolution strategies. In *Proc. 2003 Genetic and Evolutionary Computation Conf. (GECCO'03), Chicago*, Berlin, 2003. Springer.
28. R. Myers and E.R. Hancock. Empirical modelling of genetic algorithms. *Evolutionary Computation*, 9(4):461–493, 2001.
29. O. François and C. Lavergne. Design of evolutionary algorithms – a statistical perspective. *IEEE Transactions on Evolutionary Computation*, 5(2):129–148, April 2001.
30. M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

31. D.L. Whitley, K.E. Mathias, S. Rana, and J. Dzubera. Building better test functions. In L. Eshelman, editor, *Proc. of the Sixth Int. Conf. on Genetic Algorithms*, pages 239–246, San Francisco, CA, 1995. Morgan Kaufmann.
32. Stefan Droste, Thomas Jansen, and Ingo Wegener. Optimization with randomized search heuristics: The (A)NFL theorem,realistic scenarios, and difficult functions. Technical Report CI–91/00, SFB 531, Universität Dortmund, 2000.
33. D. Whitley, J.P. Watson, A. Howe, and L. Barbulescu. Testing, evaluation and performance of optimization and learning systems. Technical report, The GENITOR Research Group in Genetic Algorithms and Evolutionary Computation, Colorado State University, 2002.
34. P.J.M. van Laarhoven and E.H.L. Aarts. *Simulated Annealing: Theory and Applications*. D. Reidel Publishing Company, 1987.
35. R.C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings Sixth Symposium on Micro Machine and Human Science*, pages 39–43, Piscataway, NJ, 1995. IEEE Service Center.
36. Thomas Beielstein, Konstantinos E. Parsopoulos, and Michael N. Vrahatis. Tuning PSO parameters through sensitivity analysis. Technical Report of the Collaborative Research Center 531 *Computational Intelligence* CI–124/02, University of Dortmund, January 2002.
37. Hans-Paul Schwefel. Direct search for optimal parameters within simulation models. In *Proceedings of the twelfth annual simulation symposium*, pages 91–102. IEEE Press, 1979.
38. T. Back, T. Beielstein, B. Naujoks, and J. Heistermann. Evolutionary Algorithms for the Optimization of Simulation Models Using PVM. In J. Dongarra, M. Gengler, B. Tourancheau, and X. Vigouroux, editors, *EuroPVM '95: Second European PVM User's Group Meeting*, pages 277–282, Paris, 1995. Hermes.
39. J. Banks, J. S. Carson II, B. L. Nelson, and D. M. Nicol. *Discrete Event System Simulation*. Prentice Hall, 2001.
40. B. Naudts and L. Kallel. A comparison of predictive measures of problem difficulty in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 4(1):1–15, April 2000.
41. Hans-Paul Schwefel. *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology. Wiley Interscience, New York, 1995.
42. H.-P. Schwefel. *Evolutionsstrategie und numerische Optimierung*. Dr.-Ing. Thesis, Technical University of Berlin, Department of Process Engineering, 1975.
43. Hans-Paul Schwefel. *Numerische Optimierung von Computer–Modellen mittels der Evolutionsstrategie*, volume 26 of *Interdisciplinary Systems Research*. Birkhäuser, Basel, 1977.
44. H.-P. Schwefel. *Numerical Optimization of Computer Models*. Wiley, Chichester, 1981.
45. Thomas Bäck and Frank Hoffmeister. Adaptive search by evolutionary algorithms. In W. Ebeling, M. Peschel, and W. Weidlich, editors, *Models of Selforganization in Complex Systems (MOSES)*, pages 156–163. Akademie–Verlag, Berlin, 1992.
46. H.-G. Beyer. Zur Analyse der Evolutionsstrategien. Habilitationsschrift. University of Dortmund, 1996.
47. T. Beielstein and S. Markon. Threshold selection, hypothesis tests, and DOE methods. In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Garry Greenwood, Hitoshi Iba, Paul Marrow, and Mark Shackleton, editors, *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 777–782. IEEE Press, 2002.

48. A.M. Law and W.D. Kelton. *Simulation Modelling and Analysis*. McGraw-Hill Series in Industrial Egineering and Management Science. McGraw-Hill, New York, 3rd edition, 2000.
49. J. Kleijnen and W. Van Groenendaal. *Simulation - A Statistical Perspective*. Wiley, Chichester, 1992.
50. D. S. Johnson. A theoretician's guide to the experimental analysis of algorithms. In M. H. Goldwasser, D. S. Johnson, and C. C. McGeoch, editors, *Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges*, pages 215–250, Providence, 2002. American Mathematical Society.
51. Günter Rudolph. *Convergence Properties of Evolutionary Algorithms*. Verlag Dr. Kovač, Hamburg, October 1997.
52. J. Hyslop. A note on the accuracy of optimisation techniques. *The Computer Journal*, 15, 2, (5):140, 1972.
53. T. Beielstein, J. Dienstuhl, C. Feist, and M. Pompl. Circuit design using evolutionary algorithms. In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Garry Greenwood, Hitoshi Iba, Paul Marrow, and Mark Shackleton, editors, *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 1904–1909. IEEE Press, 2002.
54. E. D. Dolan and J. J. More. Benchmarking optimization software with performance profiles. Technical Report ANL/MCS-P861-1200, Argonne National Laboratory, 2001.
55. R. Barr, B. Golden, J. Kelly, M. Rescende, and W. Stewart. Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, 1(1):9–32, 1995.
56. Hans-Paul Schwefel. Evolutionary learning optimum–seeking on parallel computer architectures. In A. Sydow, S. G. Tzafestas, and R. Vichnevetsky, editors, *Systems Analysis and Simulation*, volume 1, pages 217–225. Akademie–Verlag, Berlin, 1988.
57. D. E. Colemann and D. C. Montgomery. A systematic approach to planning for a designed industrial experiment. *Technometrics*, 35:1–27, 1993.
58. R. Anderson. The role of experiment in the theory of algorithms. In *Proceedings of the 5th DIMACS Challenge Workshop*, volume 59 of *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, 1997.
59. I. C. Jarvie. Popper, Karl Raimund. In E. Craig, editor, *Routledge Encyclopedia of Philosophy*. Routledge, London, 1998. Retrieved November 19, 2003, from http://www.rep.routledge.com/article/DD052SECT2.
60. G. E. P. Box, W. G. Hunter, and J. S. Hunter. *Statistics for experimenters*. Wiley series in probabilty and mathematical statistics: Applied probability and statistics. Wiley, 1978.
61. G. E. P. Box and N. R. Draper. *Experimental Model Building and Response Surfaces*. Wiley, 1987.
62. J. P. C. Kleijnen. Experimental designs for sensitivity analysis of simulation models. In A. W.Heemink et al., editor, *Proceedings of EUROSIM 2001*, 2001.
63. J. M. Chambers and T. H. Hastie, editors. *Statistical Models in S*. Wadsworth & Brooks/Cole, Pacific Grove, California, 1992.
64. P. McCullagh and J.A. Nelder. *Generalized Linear Models*. Chapman and Hall, 2nd edition, 1989.
65. N. R. Draper and H. Smith. *Applied regression analysis*. Wiley series in probability and statistics. Wiley, New York, 3rd edition, 1998.
66. R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

67. D. Collett. *Modelling Binary Data*. Chapman and Hall, 1991.
68. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.
69. T. M. Therneau and E. J. Atkinson. An introduction to recursive partitioning using the rpart routines. Technical Report 61, Department of Health Science Research, Mayo Clinic, Rochester, 1997.
70. Thomas Beielstein, Sandor Markon, and Mike Preuß. Algorithm based validation of a simplified elevator group controller model. In T. Ibaraki, editor, *Proc. 5th Metaheuristics Int'l Conf. (MIC'03)*, pages 06/1–06/13 (CD–ROM), Kyoto, Japan, 2003.
71. H.H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *Comp. J.*, 3:175–184, 1960.
72. S. Markon. *Studies on Applications of Neural Networks in the Elevator System*. PhD thesis, Kyoto University, 1995.
73. E. Del Castillo. Stopping rules for steepest ascent in experimental optimization. *Communications in Statistics. Simulation and Computation*, 26(4):1599–1615, 1997.
74. D. Nolan and T. Speed. *Stat Labs – Mathematical Statistics Through Applications*. Springer, New York, 2000.