New Experimentalism Applied to Evolutionary Computation

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
der Universität Dortmund
am Fachbereich Informatik
von

Thomas Bartz–Beielstein

Dortmund

2005

Tag der mündlichen Prüfung: 20. April 2005
Dekan: Bernhard Steffen
Gutachter: Hans-Paul Schwefel, Peter Buchholz

To Eva, Leon, Benna, and Zimba.

Contents

List of Figures

List of Tables

Introduction

> The philosophy of science seems to be in a state of flux, and the possibilities opened up by the new experimentalists seem to offer genuine hope for a recovery of some of the solid intuitions of the past about the objectivity of science, but in the context of a much more detailed and articulate understanding of actual scientific practice.
>
> Robert Ackermann

This thesis tries to prepare good grounds for experimental research in evolutionary computation. We claim that experiments are necessary—a purely theoretical approach cannot be seen as a reasonable alternative. Our approach is related to the discipline of experimental algorithmics that provides methods to improve the quality of experimental research. However, this discipline is based on popperian paradigms:

1. No experiment without theory and
2. Theories should be falsifiable.

Following Hacking (1983) and Mayo (1996), we argue in this thesis that:

1. Experiment can have a life of its own and
2. Falsifiability should be complemented with verifiability.

This concept, known as the *new experimentalism*, is an influential discipline in the modern philosophy of science. It provides a statistical methodology to learn from experiments. For a correct interpretation of experimental results, it is crucial to distinguish the statistical significance of an experimental result from its scientific meaning. This thesis attempts to introduce the concept of the new experimentalism in evolutionary computation.

Research Problem

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. Sentences like "This experiment was repeated 10 times to obtain significant results" or "We have proven that algorithm A is better than algorithm B" can still be found in actual EC publications. Eiben and Jelasity (2002) explicitly list four problems:

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

In fact, there is a gap between theory and experiment in evolutionary computation. How to promote good standards and quality of research in the field of evolutionary computation was discussed during the genetic and evolutionary computation conference (GECCO) in 2002. Bentley noted:

> Computer science is dominated by the need to publish, publish, publish, but sometimes this can happen at the expense of research. All too often poor papers, clumsy presentations, bad reviews or even bad science can clutter a conference, causing distractions from the more carefully prepared work (Bentley, 2002).

There is a great demand for these topics as one can see from the interest in tutorials devoted to these questions during two major conferences in evolutionary computation, the congress on evolutionary computation (CEC) and GECCO (Wineberg and Christensen, 2004; Bartz-Beielstein and Markon, 2004).

Background

Evolutionary computation shares these problems with other scientific disciplines such as simulation, artificial intelligence, numerical analysis or industrial optimization (Dolan and More, 2001). Cohen's survey of 150 publications from the Proceedings of the Eighth American Association for Artificial Intelligence (AAAI) "gave no evidence that the work they described has been tried out on more than a single example problem" (Cohen et al., 2000). He clearly demonstrated that there is no essential synergy between experiment and theory in these papers.

But Cohen did not only report these negative results, he also provided valuable examples how empirical research can be related to theory (Cohen, 1995). Solutions from other disciplines that have been applied successfully for many years might be transferable to evolutionary computation. We have chosen four criteria to classify existing experimental research methodologies that have a lot in common with our approach. First, we can mention effective approaches. They find a solution, but are not very efficient and are not focused on understanding. Greedy, or brute-force approaches belong to this group. Second, auto-pilot methods can be mentioned. Meta-algorithms belong to this category. They might locate good parameter sets, though without providing much insight as how sensitive performance is to parameter changes. Third, approaches that model problems of mostly academic interest can be listed. These approaches consider artificial test functions or infinite population sizes. Finally, the fourth category comprehends approaches that might be applicable to our problems although they have been developed with a different goal. Methods for deterministic computer experiments can be mentioned here. We will give a brief overview of literature on experimental approaches from these four domains.

Effective Approaches

The methodology presented in this thesis has its origins in statistical *design of experiments* (DOE). But, classical DOE techniques as used in agricultural or industrial optimization must be adapted if applied to optimization models since stochastic optimization uses pseudo-random numbers (Fisher, 1935). Randomness is replaced by pseudo-randomness. For example, blocking and randomization, important techniques to reduce the systematic influence of different experimental conditions, are unnecessary in computer-based optimization. The random number seed is the only random element during the optimization run.

Classical DOE techniques are commonly used in simulation studies—a whole chapter in a broadly cited textbook on simulation describes experimental designs (Law and Kelton, 2000). Kleijnen demonstrated how to apply DOE in simulation (Kleijnen, 1987, 1997). As simulation is related to optimization (simulation models equipped with an objective function define a related optimization problem), we suggest the use of DOE for the analysis of optimization problems and algorithms (Kelton, 2000).

This thesis is not the first attempt to use classical DOE methods in EC. However, our approach takes the underlying problem instance into account. Therefore, we do not try to draw any problem independent conclusions such as: "The optimal mutation rate in genetic algorithms is 0.1." In addition, we propose an approach that requires a small amount of fitness function evaluations only. Schaffer et al. (1989) propose a complete factorial design experiment that requires 8400 run configurations, each configuration was run to $10,000$ fitness function evaluations. Feldt and Nordin (2000) use statistical techniques for designing and analyzing experiments to evaluate the individual and combined effects of genetic programming (GP) parameters. Three binary classification problems are investigated in a total of seven experiments consisting of 1108 runs of a machine code genetic programming system. Myers and Hancock (2001) present an empirical modeling of genetic algorithms. This approach requires $129,600$ program runs. François and Lavergne (2001) demonstrate the applicability of generalized linear models to design evolutionary algorithms. Again, data sets of size 1000 or even more are necessary, although a simplified evolutionary algorithm with two parameters only is designed. Myers and Hancock (2001) used a similar approach.

As we include methods from computational statistics, our approach can be seen as an extension of these classical approaches. Furthermore, classical DOE approaches rely strongly on hypothesis testing. The reconsideration of the framework of statistical hypothesis testing is an important aspect in our approach.

Auto-Pilot Methods

The search for useful parameter settings of algorithms itself is an optimization problem. Obviously optimization algorithms, so called meta-algorithms, can be defined to accomplish this task. Meta-algorithms for evolutionary algorithms have been proposed by many authors (Bäck, 1996; Kursawe, 1999). But this approach does not solve the original problem because it requires the determination of a parameter setting of the meta-algorithm. Rardin and Uzsoy (2001) present a tutorial that discusses the experimental evaluation of heuristic search algorithms when the complexities of the problem do not allow exact solutions. Their tutorial describes how to design test instances, how to measure performance, and how to analyze and present the experimental results. The main interest of their experimental evaluation approach is how close an algorithm comes to producing an optimal solution, which can be measured as the algorithm-to-optimal ratio.

Birattari et al. (2002) developed a “racing algorithm” for configuring meta-heuristics, that combines blocking designs, non-parametric hypothesis testing and Monte Carlo methods. The aim of their work was “to define an automatic hands-off procedure for finding a good configuration through statistical guided experimental evaluations”. This is unlike the approach presented here, that provides means for understanding algorithms’ performance (data-scopes similar to microscopes in biology and telescopes in astronomy) and does not claim to provide an automatic procedure to find improved configurations.

Instead, we argue that the experimenter’s skill plays an important role in this analysis. It cannot be replaced by automatic rules. The difference between automatic rules and learning tools is an important topic discussed in the remainder of this thesis.

Academic Approaches

Experimental algorithmics offer methodologies for the design, implementation, and performance analysis of computer programs for solving algorithmic problems (Demetrescu and Italiano, 2000; Moret, 2002). McGeoch (1986) examines the application of experimental, statistical, and data analysis tools to problems in algorithm analysis. Barr and Hickman (1993) and Hooker (1996) tackle the question how to design computational experiments and how to test heuristics. Johnson et al. (1989, 1991) is a seminal study of simulated annealing. Most of these studies are focused on *algorithms*, and not on *programs.* Algorithms can be analyzed on a sheet of paper, whereas the analysis of programs requires real hardware. The latter analysis includes the influence of rounding errors or limited memory capacities. We will use both terms simultaneously, because whether we refer to the algorithm or the program will be clear from the context.

Compared to these goals, our aim 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 Beielstein et al. (2003a) required more than a full week of round-the-clock computing in a batch job processing system to test 80 configurations. Furthermore, our methods are applied to real computer programs and not to abstract algorithms. However, there is an interesting link between algorithms and programs or complexity theory and the experimental approach:

Example 1 (Hooker, 1994)
Consider a small subset of very special traveling salesperson problems T. This subset is NP*-complete, and any class of problems in* NP *that contains T is* ipso facto NP*-complete. Consider the class P' that consists of all problems in* P *and T. As P' contains all easy problems in the world, it seems odd to say that problems in P' are hard. But P' is no less* NP*-complete than* TSP. *Why do we state that TSP is hard? Hooker suggests that “we regard TSP as a hard class because* we in fact find problems in TSP to be hard in practice”. *We acknowledge that TSP contains many easy problems, but we are able to generate larger and larger problems that become more and more difficult. Hooker suggests that it is this empirical fact that justifies our saying that TSP contains characteristically hard problems. And, in contrast to P', TSP is a natural problem class, or as philosophers of science would say, a natural kind.* ■

Approaches With Different Goals

Although our methodology has its origin in DOE, classical DOE techniques used in agricultural and industrial simulation and optimization tackle different problems and have different goals.

Parameter control deals with parameter values (endogenous strategy parameters) that are changed during the optimization run (Eiben et al., 1999). This differs from our approach that is based on parameter values that are specified before the run is performed (exogenous strategy parameters). The assumption that specific problems require specific EA parameter settings is common to both approaches (Wolpert and Macready, 1997).

Design and analysis of computer experiments (DACE) as introduced in Sacks et al. (1989) models the deterministic output of a computer experiment as the realization of a stochastic process. The DACE approach focuses entirely on the correlation structure of the errors and makes simplistic assumptions about the regressors. It describes "how the function behaves", whereas regression as used in classical DOE describes "what the function is" (Jones et al., 1998, p.14). DACE requires other experimental designs than classical DOE, e.g. Latin hypercube designs (McKay et al., 1979). We will discuss differences and similarities of these designs and present a methodology how DACE can be applied to stochastic optimization algorithms.

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.

Common Grounds: Optimization Runs Treated as Experiments

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.
Optimization runs will be treated as experiments. In our approach, an experiment consists of a problem, an environment, an objective function, an algorithm, a quality criterion, and an initial experimental design. We will use methods from computational statistics to improve, compare and understand algorithms' performances. The focus in this work lies on natural problem classes: Its elements are problems that are based on real-world optimization problems in contrast to artificial problem classes (Eiben and Jelasity, 2002). 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.

Furthermore, the methodology presented in this thesis 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. They are applicable to generate systematically feasible starting points that are better than randomly generated initial points, or to guide the optimization process to promising regions of the search space. Meta-model assisted search strategies as proposed in Emmerich et al. (2002) can be mentioned in this context. Jin (2003) gives a survey over approximation methods in EC.

Before introducing our understanding of experimental research in EC, we may ask for the importance of experiments in other scientific disciplines. For example, the role of experiments in economics changed radically during the last decades.

Wind Tunnels

The path breaking work of Vernon L. Smith (2002 Nobel Prize in Economics together with Daniel Kahneman) in experimental economics provided criteria to find out whether economic theories hold up in reality. Smith demonstrated that a few relatively uninformed people can create an efficient market. This result did not square with theory. Economic theory claimed that one needed a horde of "perfectly informed economic agents". He reasoned that economic theories could be tested in an experimental setting: An economic wind tunnel. Smith had a difficult time to get the corresponding article published (Smith, 1962). Nowadays this article is regarded as the landmark publication in experimental economics.

Today, many cases of economic engineering are of this sort. For example, before being exported to the real world, the auctions for "third generation" mobile phones were designed and tested in the economic laboratory at CalTech (Guala, 2003). This course of action suggests that experiments in economics serve the same function that a wind tunnel does in aeronautical engineering. But, the relationship between the object of experimentation and the experimental tool is of importance: How much reductionism is necessary to use a tool for an object? Table 1 lists some combinations. Obviously some combinations fit very good, whereas others make no sense at all.

We propose an experimental approach to analyze algorithms that is suitable to discover important parameters and to detect superfluous features. But before we can draw conclusions from experiments we have to take care that the experimental results are correct. We have to provide means to control the error, because we cannot ensure that our results are always sound. Therefore the concept of the new experimentalism is regarded next.

The New Experimentalism

The new experimentalism is an influential trend in recent philosophy of science that provides statistical methods to set up experiments, to test algorithms, and to learn from the resulting errors and successes. The new experimentalists are seeking a relatively secure basis for science, not in theory or observation but in experiment. To get the apparatus working for simulation studies is an active task. Sometimes the recognition of an oddity leads to new knowledge. Important representatives of the new experimentalism are Hacking (1983), Galison (1987), Gooding (1989), Mayo (1996), and Franklin (2003). Deborah Mayo, whose work is in the epistemology of science and the philosophy of statistical inference, proposes detailed way in which scientific claims are validated by experiment. A scientific claim can only be said to be supported by experiment if it passes a severe test. A claim would be unlikely to pass a severe test if it were false. Mayo developed methods to set up experiments that enable the experimenter, who has a detailed knowledge of the effects at work, to learn from error.

Overview

This thesis develops a solid statistical methodology, which we consider to be essential in performing computer experiments. New concepts for an objective interpretation of experimental results are introduced in this thesis. It consists of seven chapters (Experiments, Statistics, Problems, Designs, Search, Comparison, and Understanding) and a concluding discussion. Each chapter closes with a summary of the key points.
The first chapter introduces the concept of the new experimentalism for computer experiments

Table 1: *Relationship between experimental objects and experimental tools. Some combinations, for example reality – computer, require some kind of reductionism. Others, for example algorithm – wind tunnel, are useless.*

object of experimentation	experimental tool
reality	computer
reality	thought experiment
reality	wind tunnel
airplane	computer
airplane	thought experiment
airplane	wind tunnel
algorithm	computer
algorithm	thought experiment
algorithm	wind tunnel

and discusses central elements of an understanding of science. It details the difference between demonstrating and understanding, and between significant and meaningful. To incorporate these differences, separate models are defined: Models of hypotheses, models of experimental tests, and models of data. This leads to a re-interpretation of the Neyman-Pearson theory of testing (NPT). Since hypothesis testing can be interpreted objectively, tests can be considered as learning tools. Analyzing the frequency relation between the acceptance (and the rejection) of the null hypothesis and the difference in means enables the experimenter to learn from errors. This concept of learning tools provides means to extend Popper's widely accepted claim that theories should be falsifiable.
Statistical definitions for Monte Carlo methods, classical design and analysis of experiments, tree based regression methods and modern design and analysis of computer experiments techniques are given in the second chapter. A bootstrap approach that enables the application of learning tools if the sampling distribution is unknown is introduced. This chapter is rather technical because it summarizes the relevant mathematical formulas.
Computer experiments as discussed in this thesis are conducted to improve and to understand the algorithm's performance. Chapter 3 presents optimization problems from evolutionary computation that can be used to measure this performance. Before an elevator group control problem is introduced as a model of a typical real-world optimization problem, some commonly used test functions are presented. Problems related to test suites are discussed as well.
Different approaches to set up experiments are discussed in Chapter 4. Classical and modern designs for computer experiments are introduced. A sequential design based on DACE that maximizes the expected improvement is proposed.
Search algorithms are presented in Chapter 5. Classical search techniques, for example the Nelder Mead "simplex" algorithm, are presented as well as stochastic search algorithms. The focus lies on particle swarm optimization algorithms, that build a special class of bio-inspired algorithms.
The discussion of the concept of optimization provides the foundation to define performance measures for algorithms in Chapter 6. A suitable measure reflects requirements of the optimization scenario or the experimental environment. The measures are categorized with respect to effectivity and efficiency. Now, the necessary components according to the discussion in the previous chapters to perform computer experiments are available: A problem,

an environment, an objective function, an algorithm, a quality criterion, and an experimental design. After summarizing a classical DOE approach of finding better suited exogenous parameters (tuning), a sequential approach that comprehends methods from computational statistics is presented. To demonstrate that our approach can be applied to any arbitrary optimization algorithm, several variants of optimization algorithms are tuned. Tools from error statistics are used to decide whether statistically significant results are scientifically meaningful.
Chapter 7 closes the circle opened in Chapter 1 on the discussion of testing as an automatic rule and as a learning tool. Provided with the background from Chapter 1, the aim of Chapter 7 is to propose a method to learn from computer experiments and to understand how algorithms work. Various schemes for selection under noise for direct search algorithms are presented. Threshold selection is related to hypothesis testing. It serves as an example to clarify the difference between tests as rules of inductive behavior and tests as learning tools. A summary and discussion of important results concludes this thesis.

Introducing the new experimentalism in evolutionary computation provides tools for the experimenter to understand algorithms and their interactions with optimization problems. Experimentation is understood as a means for testing hypotheses, the experimenter can learn from error and control the consequences of his decisions. The methodology presented here is based on the statistical methods most widely used by today's practicing scientists. It might be able "to offer genuine hope for a recovery of some of the solid intuitions of the past about the objectivity of science"(Ackermann, 1989).

Acknowledgments

Before we go into *medias res*, I would like to acknowledge the support of many people who made this thesis possible.
First and foremost, Hans-Paul Schwefel, the head of the Chair of Systems Analysis for providing a cooperative and stimulating work atmosphere. His thoughtful guidance and constant support in my research were very valuable and encouraging.
I am also thankful to Peter Buchholz for his kindness in being my second advisor.

Thomas Bäck supported my scientific research for many years, beginning at the time when I was a student and working at the Chair of Systems Analysis and during the time I did work for NuTech Solutions. He also established the contact to Sandor Markon which resulted in an inspiring collaboration devoted to questions related to elevator group control and the concept of threshold selection. Sandor Markon also provided guidance in Korea and Japan, which made my time there very enjoyable.

I appreciated the discussions relating to threshold selection with Dirk Arnold very much. They built the cornerstone for a productive research that is devoted to selection and decision making under uncertainty.

The first official presentation of the ideas from this thesis during the CEC tutorial on experimental research in evolutionary computation in 2004 was based on the collaboration and the helpful discussions with Mike Preuss. Tom English's support during the preparation and presentation of this tutorial were very comforting. I also very much enjoyed the constructive exchange of information with the people from the EC "task force", Steffen Christensen, Gwenn Volkers and Mark Wineberg.

My colleagues Boris Naujoks, Karlheinz Schmitt and Christian Lasarczyk shared their knowledge and resources, helped in many discussions to clarify my ideas and made the joint work a very fortunate experience. Konstantinos E. Parsopoulos and Michael N. Vrahatis aroused my interest in particle swarm optimization. Discussions with students, especially with Christian Feist, Marcel deVegt and Daniel Blum have been a valuable source of inspiration during this research.

Thomas Bartz-Beielstein
Dortmund, February 2005

Chapter 1

Experiments

> The physicist George Darwin used to say that every once in a while one should do a completely crazy experiment, like blowing the trumpet to the tulips every morning for a month. Probably nothing would happen, but what if it did?
>
> Ian Hacking

1.1 Introduction

This chapter presents an attempt to justify the role of experiments in evolutionary computation. First, problems related to experiments are presented. Objections stated by theoreticians like “Algorithms are formal objects and should be treated formally” are discussed. After considering these objections, we present an experimental approach in evolutionary computation. Important goals for scientific research in evolutionary computation are proposed. Experimental algorithmics is an influential discipline that provides widely accepted methods to tackle these scientific goals. It is based on a popperian understanding of science. After introducing the concept of models in science, the new experimentalism is presented. It goes beyond Popper’s concept that only results that are falsifiable should be treated as scientific. The new experimentalists claim that the experimenter can learn from experiments. Mayo introduced an approach based on the Neyman-Pearson theory of statistical testing that enables the experimenter to perform experiments in an objective manner. It is important to note that statistically significant results are not automatically meaningful. Therefore some space must be left between the statistical result and its scientific import. Finally, the relationship between theory and practice is reconsidered.

The modern theory of statistical testing presented in this chapter is based on Hacking (1983) and Mayo (1996).

1.2 The Gap between Demonstrating and Understanding

We first start with a comparison of two parameterizations of a stochastic global optimization method. This comparison is based on real optimization data, but it is kept as simple as possible for didactical purposes.

Example 1.1 (PSO swarm size)
Analyzing a particle swarm optimization algorithm *(PSO), we are interested in testing whether or not the swarm size has a significant influence on the performance of the algorithm. The 10 dimensional Rosenbrock function was chosen as a test function. Based on the parameterization in Shi and Eberhart (1999), the swarm size was set to* 20 *and* 40. *The corresponding settings will be referred to as run PSO(20) and PSO(40) respectively. The question is whether the increased swarm size improves the performance of the PSO. As in Shi and Eberhart (1999), a random sample is drawn from each of the two populations. The average performance* $\overline{y}_1$ *of* $n = 50$ *runs of PSO(20) is* 108.02, *whereas the average performance* $\overline{y}_2$ *of* $n = 50$ *runs of PSO(40) is* 56.29. *The same number of function evaluations was used in both settings. The number of runs* n *is referred to as the* sample size, *and* $\overline{y}$ *denotes the* sample mean. ■

Example 1.1 demonstrates at first sight, that the hypothesis (H)

(H-1.1) PSO(40) outperforms PSO(20)

is correct. But can we really be sure that (H-1.1) is true? Is this result statistically significant? Are the influences of other parameters on the algorithm's performance negligible? Are 50 repeats sufficient? How does the run-length, that is the number of iterations, influence the result? However, even if we assume that (H-1.1) is correct, what can we learn from this conclusion?

As will be demonstrated later on and as some readers already know, choosing a suitable parameterization enables the experimenter to demonstrate anything—algorithm A is better than algorithm B or the other way round.

The remainder of this thesis deals with questions related to these problems and provides a methodology to perform comparisons in a statistically sound manner.

1.2.1 Why Do We Need Experiments in Computer Science?

It is a difficult task to set up experiments correctly. Experimental results may be misleading. So one may ask why to perform computer experiments at all.

There are theoretical and empirical approaches to study the performance of algorithms. In contrast to some researchers who consider merely the former as scientific, many practitioners are convinced that the theoretical approach alone is not well-suited to judge an algorithm's performance.

Why is the empirical work viewed as unscientific? One reason might be the lack of standards for empirical methods. Empirical work is sometimes considered as "lowbrow or unsophisticated"(Hooker, 1994). Additionally, the irreproducibility of the results discredits empirical approaches (Eiben and Jelasity, 2002). But these are problems that can be mastered, at least in principle. The main objection against empirical work lies deeper. Hooker hits the nail on the head with the following characterization: The main objection against empirical work is comparable to the uneasiness that arises when "verifying that opposite interior angles are equal by measuring them with a protractor"(Hooker, 1994). This can be formulated as

Statement 1
Algorithms are defined as formal systems and should be studied with formal methods.

Reasoning that many founders of modern science like Descartes, Leibniz, or Newton studied formal systems with empirical methods does not give a completely satisfactory response to this objection. After discussing the role of models in science we will reconsider this objection and give a well founded answer that uses some fundamental concepts from the philosophy of science. However, even under the assumption that Statement 1 is true, studying formal systems is not as trivial as it might appear at first sight. Severe objections arise when Statement 1 is considered in detail. The construction of a self-explanatory formal system requires a huge complexity. These systems cannot be applied in real-life situations (Mertens, 1990). This may be one reason for the failure of the enthusiastically propagated set theoretical approach to mathematics in primary schools in the 1960s. Nowadays it is widely accepted that the Peano-axioms do not provide a suitable context to introduce the system of whole numbers for primary schools (Athen and Bruhn, 1980).
But not only its complexity makes the formal approach difficult. As Statement 1 cannot be proven, it is rather subjective. It is obvious that:

1. Algorithms treated as formal systems require some kind of reductionism.
2. Reductionism works in some cases, but fails in others.

Based on our subjective experience as experimenters we can claim that:

Statement 2
Reductionism often fails in algorithmic science.

Hooker gives an example that emphasizes the importance of the right level of reductionism or abstraction that provides an understanding of the underlying phenomena: Investigating the behavior of algorithms with formal methods is like applying quantum physics to geology to understand plate tectonics. Even if one can in principle deduce what the algorithms are going to do, we would not understand why they behave as they do (Hooker, 1994). As will be seen in the following, the concept of model plays a central role in tackling these problems.

Comparing mathematical models and experiments, the following statements are true:

1. Results from mathematical models are *more* certain than results from experiments.
2. Results from mathematical models are *less* certain than results from experiments.

As the conclusions must follow from the premises, mathematical models are more certain. This justifies the first statement. However, as these premises are more hypothetical and arbitrary, the conclusions are less certain. This confirms the second statement. Both, mathematical models and experiments, deliver only hypotheses. Or, as stated by Einstein: "As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality" (Newman, 1956).

Summarizing, one can claim that a solely theoretically oriented approach is not completely satisfactory. We would like to mention a few more reasons why experiments are useful:

- Theories may be incomplete, they may have holes. Consider the Nelder-Mead simplex algorithm (Nelder and Mead, 1965), one of the most popular methods for nonlinear unconstrained optimization. "At present there is no function in any dimension greater than 1 for which the original Nelder Mead algorithm has been proved to converge to a minimizer" (Lagarias et al., 1998).

- Observations and experiments may suggest new theories. Existing theories can be tested and refined by experiments. The 2004 NASA gravity probe B mission can be mentioned here. Gravity probe B is the relativity gyroscope experiment being developed by NASA and Stanford University to test two extraordinary, unverified predictions of Einstein's general theory of relativity.

- Experiments can bridge the gap between theory and practice: Experiences from the collaborative research center "Design and Management of Complex Technical Processes and Systems by Means of Computational Intelligence Methods" in Dortmund show that the engineer's view of complexity differs from the theoretician's view (Schwefel et al., 2003).

- Digital computers use a finite number of decimal places to store data. To check theoretically derived convergence results, computer experiments that consider rounding errors, have to be performed (Schwefel, 1995).

- Worst case $\neq$ average case: As worst case scenarios can theoretically easier be analyzed than average case scenarios, many theoretical results are related to the former. But the real world is "mostly" an average case scenario (Briest et al., 2004).

- To obtain average case results, it is necessary to define a probability distribution over randomly generated problem instances. But, this distribution is "typically unreflective of reality" (Hooker, 1994). Real-world optimization problems are not totally random, they possess some kind of structure: To model this structure, Kan (1976) introduced job-correlated processing times for scheduling problems. As a consequence, theoretical lower bound calculations that perform very well for randomly generated problem instances, provide "an extremely poor bound on job correlated problems" (Whitley et al., 2002). We will come back to this problem in Chapter 3.

- And last, but not least: What is c in $O(n) + c$?

Many theoreticians would accept the point of view that experiments are useful—at least when they support theories.
If experiments are elements of science—this point has not been clarified yet—an experimental methodology to deal with important research questions is needed. Already ten years ago, Hooker (1994), coming from the operations research community, postulated to "build an empirical science of algorithms". The key ingredients of his empirical science are statistical methods and empirically based theories that can be submitted to rigorous testing. We claim that the time is ripe to transfer these ideas to the field of evolutionary algorithms.
First, we will ask "what is the role of science?" Or to be more concrete: "Which are important research topics especially in evolutionary computation?"

1.2.2 Important Research Questions

We claim that important elements of research in evolutionary computation comprise tasks like:

Investigation: Specifying optimization problems and analyzing algorithms. Which are important parameters, what should be optimized?

Comparison: Comparing the performance of competing search heuristics such as evolutionary algorithms, simulated annealing, and 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.

Quality: Improving the robustness of the results obtained in optimization or simulation runs. Robustness includes insensitivity to exogenous factors that can affect the algorithms' performance, and minimization of the variability around the solutions obtained (Montgomery, 2001).

These four goals are also discussed in the discipline of experimental algorithmics. As it has gained much attention in the last years, we will present this approach to establish experiments as scientific means first. (Pre-)experimental planning is of importance in experimental algorithmics.

1.3 Experimental Algorithmics

1.3.1 Pre-experimental Planning

Pre-experimental planning has a long tradition in other scientific disciplines. For instance, Colemann and Montgomery (1993) present a checklist for the pre-experimental planning phases of an industrial experiment. It covers the following topics: Objectives of the experiment, a performance measure, 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 Anderson (1997). Good empirical work must pass the following tests: "It must be both convincing and interesting" (Anderson, 1997). Moret (2002) gives a suitable 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 in the model. This is one example for blowing the trumpet to the tulips.

1.3.2 Guidelines from Experimental Algorithmics

As we have classified important parameters of the algorithm to be analyzed, and have defined a measure for its performance, we can conduct experiments to assess the significance of single parameters such as population size or selective pressure. 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. Guidelines (GL) from experimental algorithmics to set up and to perform experiments read as follows (Moret, 2002):

(GL-1) Question: State a clear set of objectives. Formulate a question or a hypothesis. Typical questions or hypotheses read: "Is the selective pressure $\nu = 5$ a good choice for the optimization problem under consideration?", or "PSO works better when the swarm size is ten times the dimension of the search space compared to a parameterization that uses a fixed swarm size."

(GL-2) Data collection: After an experimental design is selected, simply gather data. Do not modify the hypothesis until all data have been collected.

(GL-3) Analysis: Analyze the data to test the hypothesis stated above.

(GL-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 complies 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" (Jarvie, 1998). Or, as Sanders introduces the related discipline of algorithm engineering: "Algorithm engineering views design, analysis, implementation, and experimental analysis of algorithms as an integrated process in the tradition of Popper's scientific method" (Sanders, 2004). Figure 1.1 depicts a commonly accepted view on the relationship between theory and experiment. This position is nowadays broadly accepted in the computer science community. However, it is intensely discussed in the philosophy of science. Results from these discussions have a direct impact on the experimental methodology in evolutionary computation. Therefore we will present the fundamental ideas in the following. After introducing the framework of the new experimentalists in Section 1.6, Popper's position will be reconsidered.

To introduce the concept of models as a central element of science, we describe an inherent problem of nearly any real-world situation: Noise.

1.4 Observational Data and Noise

Even if a scientific hypothesis or claim describes a phenomenon of investigation correctly, observational data must not precisely agree with it. The accuracy and precision of such data may be limited by measurement errors or inherent fluctuations of the response quantity, for example turbulences. Another source of distortion may lie in the inherent probabilistic nature of the scientific hypotheses. Moreover, the observational data are discrete in contrast to scientific hypotheses that may refer to continuous values (Mayo, 1983).

Although computer programs are executed deterministically, evolutionary computation has to cope with noise. Stochastic optimization uses pseudo-random numbers. Randomness is replaced by pseudo-randomness.

As common or antithetic seeds can be used, the optimization practitioner has much more

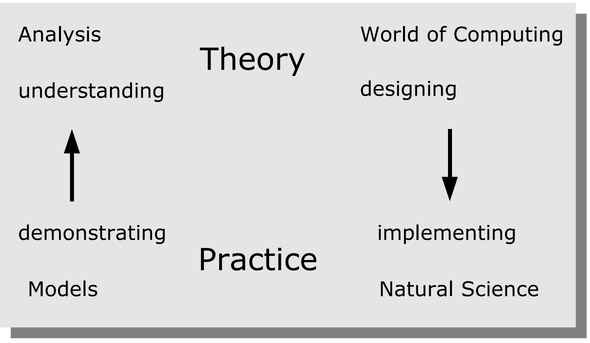


Figure 1.1: *A first approach to model the relationship between theory and practice. Practice can benefit from theory and vice versa. Demonstrating good results is only the first step in the scientific process, whereas nature's reality can be seen as the judge of a scientific theory. Source: Bartz-Beielstein (2003).*

control over the noise in the experiments and can control the source of variability (Kleijnen, 1997). The different optimization runs for one specific factor combination can be performed under exactly the same conditions—at least in principle: Even under exactly the same conditions different hardware can produce unexpected results. To compare different run configurations under similar conditions *variance-reduction techniques* (VRT) such as *common random numbers* (CRN) and antithetic variates can be applied (Law and Kelton, 2000).

Random error or noise can be classified as an unsystematic effect. Systematic errors, for example the selection of a wrong regression model, are referred to as *bias*. Santner et al. (2003) distinguish control, noise (or environmental) variables, and model variables. Control variables can be set by the experimenter to control the output (response) of an experiment, noise variables depend on the environment.

Statistical methods can be used to master problems due to noise. They require the specification of models. In the following section, the model concept from mathematical logic is complemented with a model concept that defines models as tools for representing and understanding the world.

1.5 Models

Models are central elements of an understanding of science. Giere (1999) concludes that "models play a much larger role in science than even the most ardent enthusiasts for models have typically claimed". Perhaps the most influential paper that describes the meaning and use of models in mathematics (especially in mathematical logic) and empirical sciences is Suppes (1969a). Based on Tarski's definition: "A possible realization in which all valid sentences of a theory T are satisfied is called a model of T" (Tarski, 1953), Suppes asserts "that the meaning of the concept of model is the same in mathematics and the empirical science", although the concept of model is used in a different manner in these disciplines (Suppes, 1969b). The concept of model used by mathematical logicians is the basic and fundamental concept of model needed for an exact statement of any branch of empirical science.

Logicians examine models that consist of abstract entities, i.e. geometrical objects. Suppes' proposition that there is no difference in the concept of a model in empirical science and in mathematics is based on the consideration that these objects could be physical objects. Or as David Hilbert has already stated decades ago: "Man muss jederzeit anstelle von Punkten, Geraden und Ebenen Tische, Stühle und Bierseidel sagen können".[1] Suppes establishes a relationship between theories (sets of axioms) and models (sets of objects satisfying the axioms). A model provides an interpretation of a set of uninterpretated axioms, called interpretative models.

Introducing error terms, model descriptions such as mathematical formulas, can be interpreted as hypotheses about real-world systems. Hypotheses can be tested based on evidence obtained by examining real-world objects, i.e. by performing experiments. Higher level models are not compared directly with data, but with models of data that rank lower in the hierarchy of models. In between must be a model of experiments (Suppes, 1969b). Giere summarizes Suppes' hierarchy of models as follows:

[1]This famous quotation cannot be found in Hilbert's publications. Walter Felscher wrote: "I have looked through Hilbert's articles on geometry, as well as through those on the foundations of mathematics, but nowhere did I find formulations mentioning Tische, Stuehle, Bierseidel. So the dictum seems indeed to be only such, not a quotation documentable in Hilbert's own publications"(Felscher, 1998).

1. Theoretical principles.
2. Theoretical models.
3. Models of experiments.
4. Models of data.
5. Data.

Theoretical models describe how a substantive inquiry can be divided into local questions that can be probed. Experimental models are used to relate questions to canonical questions about the particular type of experiment and how to relate data to these experimental questions. Models of data describe how raw data can be generated and modeled so as to put them into a canonical form. In addition they describe how to check if the data generation satisfies assumptions of the experimental models.

Following Suppes and Hilbert, models consist of abstract entities that could be *in principle* physical objects. But is this view of models adequate for physical objects? Giere (1999) discusses maps, diagrams, and scale models (models of the solar system or model houses) as representational models. He characterizes Suppes' models as *instantial* in contrast to his understanding that is *representational*. Representational means that models are tools for representing the world for specific purposes, and not primarily providing means for interpreting formal systems. The representational view is related to the systems analysis process that requires a discussion of the context in which the need for a model arises before the subject of models and modeling is introduced (Schmidt, 1986).

From the instantial view of models there is a direct relationship between linguistic expressions and objects. A circle can be defined as the set of points that have a constant distance (radius) from one specific point. The mathematical object "circle" can be linked to the linguistic expression without loss. But physical objects that cannot be observed precisely, and cannot be defined as exactly as theoretical objects, require a different conception.
Testing the fit of a model with the world, the model is compared with another model of data. It is not compared to data. And, scientific reasoning is "models almost all the way up and models almost all the way down" (Giere, 1999). A significant use of models appears in mathematical statistics, where models are used to analyze the relation between theory and experimental data (Suppes, 1969a). We will concentrate on the usage of models in mathematical statistics. The following section presents models of statistical testing in the framework of the new experimentalism.

1.6 The New Experimentalism

A naive description of the popperian paradigm how scientific theories are constructed is based on three assumptions (Chalmers, 1999):

1. Generalizations are based on a huge number of observations.
2. Observations have been repeated under a huge number of varying conditions.
3. No statement violates commonly accepted principles.

The vagueness of the term “huge number” is not the only problem of this approach. Sometimes, only a small number of observations is necessary to understand an effect: The destructive power of the atomic bomb has fortunately been demonstrated only seldom in the last decades.
The second assumption requires additional knowledge to differentiate between significant and insignificant variations (the colors of the experimenter’s socks should have no significant impact on the experimental outcome—*ceteris paribus conditions* of an experiment are usually not incorporated in models). This additional knowledge can be concluded from well-known facts. As these well-known facts have to be justified and depend on the specification of further additional knowledge, this leads to an infinite regress.

Popper demands that theories should be falsifiable. “Good” theories survived many attempts of falsification. However, it remains unclear whether it is the theory or the additional assumptions, which are necessary to construct the theory, that are responsible for its falsification. Also Popper does not provide positive characterizations that would allow the discovery of survivable theories.

This is where the new experimentalism comes into play: The new experimentalists are seeking for a scientific method, not through observation (passive), but through experimentation (active). They claim that experiments have negative and positive functionality. The experimenter can learn from mistakes because he has some knowledge of their causes.

Experiments live a life of their own, they do not necessarily require complex theories and are theory-neutral. Faraday’s electric motor is one famous example to illustrate how experiments can be performed successfully and independently from high-level theories: “Faraday had no theory of what he had found” (Hacking, 1983, p. 211).

The new experimentalists are looking for scientific conclusions that can be validated independently from complex abstract theories. Experiments can verify and falsify assertions and identify formerly unknown effects. Experimental results are treated as samples from the set of all possible results that can be drawn from experiments of this type. Error-statistics are used to assign probabilities to sampling events. A theory is supported if predictions based on this theory have been proven.

Science is seen as the growth of experimental knowledge. The new experimentalists provide substantial means that enable experimenters to derive experimental knowledge independently from theory. One example how learning from experiments can be carried out will be detailed next.

1.6.1 Mayo’s Models of Statistical Testing

Mayo (1996) attempts to capture the implications of the use of models in mathematical statistics in a rigorous way. A statistically modeled inquiry consists of three components (Mayo, 1983), see Figure 1.2:

(1) A *scientific claim* $\mathcal{C}$ can be modeled in terms of *statistical hypotheses* about a population. *Statistical models of hypotheses* enable the translation of scientific claims into claims about a certain population of objects. A *random variable* Y with *probability distribution* P describes a quantity of interest. *Statistical hypotheses* are hypotheses about the value of parameters of the probability distribution P, i.e. the mean μ of Y. A *probability model*

$$M(\mu) = \{Pr(Y|\mu), \mu \in \Omega\} \tag{1.1}$$

describes Y, where Ω denotes the *parameter space.*

(2) Experimental *testing rules* can be used to model the observational analysis of the statistical hypotheses. The *sample space* $\mathcal{Y}$ is the set of possible experimental results $y = (y_1, \ldots, y_n)$. Each y_i is the realization of an independent random variable Y_i that is distributed according to $M(\mu)$. The probability of an experimental result $Pr(y|\mu)$ can be determined. Based on a *test statistic* T, a *testing rule* RU maps outcomes in $\mathcal{Y}$ to various claims about the model of hypotheses $M(\mu)$:

$$RU : \mathcal{Y} \rightarrow M(\mu). \tag{1.2}$$

A *statistical model of experimental tests* $ET(\mathcal{Y})$ is the triple $(\mathcal{Y}, Pr(x|\mu), RU)$.

(3) The actual *observation* $\mathcal{O}$ can be modeled in terms of a statistical sample from the population. A statistical model of data models an empirical observation $\mathcal{O}$ as a particular element of $\mathcal{Y}$. It includes various sample properties of interest, for example the sample average.

As we will see in the following, it is crucial to leave some room between statistical and scientifical conclusions. A statistically significant conclusion is not automatically scientifically meaningful.

1.6.2 Neyman-Pearson Philosophy

The classical *Neyman-Pearson theory of testing* (NPT) requires the determination of the region of the parameter space Ω in the hypothesis model $M(\mu)$ that will be associated with the null hypothesis H and the determination of the region that will be associated with an alternative hypothesis J. Applications of the NPT testing rules lead to a rejection of H and an acceptance of J or vice versa.

Before the sample is observed, the experimental testing model $ET(\mathcal{Y})$ specifies which of the outcomes in $\mathcal{Y}$ should be taken to reject H. These values form the *critical region* (CR). A *type I error* occurs if a true H is rejected, a *type II error*, when a false H is accepted, α and β denote the corresponding error probabilities. These error probabilities can be controlled by specifying a test statistic T and a testing rule RU that defines which of the possible values of T are mapped to the critical region. In general, a (test) statistic is a function of the observed random variables obtained in a random sample. It can be used as a point estimate for a population parameter, for example the mean, or as a test statistic in hypothesis testing. A testing rule RU with $Pr(T \in \text{CR}|\mu \in \Omega_H) \leq \alpha$ and $1 - Pr(T \in \text{CR}|\mu \in \Omega_J) \leq \beta$ can be specified, because the probability that $T \in \text{CR}$ can be determined under various values of μ. The event $\{RU \text{ rejects } H\}$ can be identified with $\{T \in \text{CR}\}$. It follows

$$Pr(RU \text{ rejects } H|H \text{ is true }\} = Pr(T \in \text{ CR } |\mu \in \Omega_H\} \leq \alpha \tag{1.3}$$

$$Pr(RU \text{ accepts } H|J \text{ is true }\} = 1 - Pr(T \in \text{ CR } |\mu \in \Omega_J\} \leq \beta. \tag{1.4}$$

The simultaneous minimization of α and β is a conflicting goal. The two types of error are inversely related to each other, it is impossible to minimize both of them simultaneously without increasing the sample size. Usually, the *significance level* α of a test is selected first. The significance level of a test can also be referred to as the *size*. In a second step a test with a small β value is chosen. The best NPT test is the one that minimizes the probability of

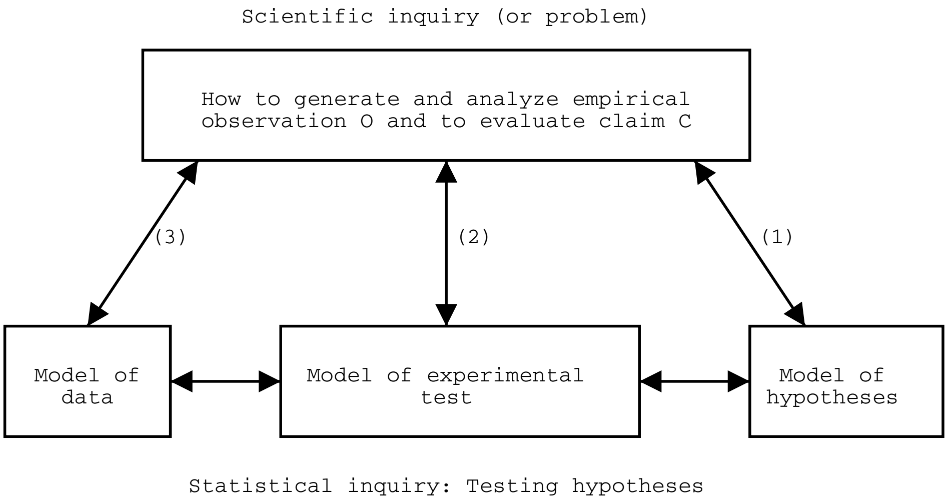


Figure 1.2: *Models of statistical testing, from Mayo (1983). It is crucial to leave some room between statistical and scientifical conclusions. A statistically significant conclusion is not automatically scientifically meaningful.*

type II errors for all possible values of μ under the alternative J. NPT tests are objective, because they control the error probabilities independently from the true value of μ.

It is fundamental to be aware of two different interpretations of statistical tests. In a similar manner as Cox and Hinkley distinguish "the theory of statistical methods for the interpretation of scientific and technological data" and statistical decision theory, "where statistical data are used for more or less mechanical decision making"(Cox and Hinkley, 1974), Mayo distinguishes rules of inductive behavior and learning tools.

Rules of inductive behavior Statistical tests can be interpreted as *rules of inductive behavior* and provide an automatic rule for testing hypotheses (Mayo, 1996, p. 368). "To accept a hypothesis H means only to decide to take action A rather than action B"(Neyman, 1950, p. 258). These behavioristic models of tests and the related automatisms are adequate means "when circumstances force us to make a choice between two courses of action"(Neyman, 1950, p. 258). Automatic test rules play an important role for selection procedures of search algorithms under uncertainty, for example the threshold selection scheme for direct search algorithms introduced later on.

Learning tools Mayo (1983) reformulates the Neyman-Pearson theory of testing and argues that it provides an objective theory of statistics. The control of error probabilities provides means to evaluate what has been learned from the results of a statistical test. Mayo describes tests as *learning tools*: "A test teaches about a specific aspect of the process that produces the data" (Mayo, 1996, p.382).

It might be useful to present an example that uses simplified assumptions, e.g. common known variances, to explain the objective theory of statistical testing. The reader should be acquainted with the basics of statistical testing. A short introduction is given in Chapter 2.

Example 1.2 (PSO swarm-size)
In Example 1.1 a random sample was drawn from each of the two populations to determine whether or not the difference between the two population means *is equal to δ. The two*

samples are independent, each population is assumed to be normally distributed with common known standard deviation σ. The question is whether the increased swarm size improves the performance of the PSO. This can be formulated as the scientific claim $\mathcal{C}_1$:

PSO(40) has a better (smaller) mean best fitness value than PSO(20).

(A) *Statistical hypothesis: The model $M(\delta)$ is in the class of normal distribution, that is: It is supposed that the standard deviation σ is known ($\sigma = 160$) and that the variability of $\overline{Y}_i$, the mean best fitness value from n experiments, can be modeled by a normal distribution, $i = 1, 2$. If PSO(40) does not improve the performance, the difference δ between the two population means μ_1 and μ_2 would be zero. On the other hand, if $\mathcal{C}_1$ is true, δ will be greater than zero. The hypotheses read:*

$$\begin{aligned} &\textit{Null hypothesis } H: \delta = 0 \textit{ in } \mathcal{N}(\delta, \sigma^2) \\ &\textit{Alternative hypothesis } J: \delta > 0 \textit{ in } \mathcal{N}(\delta, \sigma^2). \end{aligned} \tag{1.5}$$

(B) *Specifying the components of an experimental test (ET_1): The vector $y_i = (y_{i1}, \ldots, y_{in})$ represents n observations from the i-th configuration, $\overline{y}_i$ denotes the i-th sample mean, $i = 1, 2$. The experimental test statistic is $T = \overline{Y}_{12} = \overline{Y}_1 - \overline{Y}_2$, its distribution under H is $\mathcal{N}(0, 2\sigma^2/n)$.* The upper α percentage point of the normal distribution *is denoted as z_α, for example $z_{0.05} = 1.64$ or $z_{0.01} = 2.33$. As the number of observations was set to $n = 50$, it follows that the value of the* standard error *is $\sigma_{\overline{d}} = \sigma_{\overline{y}_1 - \overline{y}_2} = 160\sqrt{2/50} = 32$. The significance level of the test was $\alpha = 0.01$, thus $z_\alpha = z_{0.01} = 2.33$. So the test rule RU is*

$$T: \textit{ Reject } H: \delta = 0 \textit{ if } T = \overline{Y}_1 - \overline{Y}_2 \geq 0 + z_\alpha \cdot \sigma_{\overline{d}} = 74.44.$$

(C) *Sample data: The average performance $\overline{y}_1$ of $n = 50$ runs of PSO(20) is 108.02, whereas the average performance $\overline{y}_2$ of $n = 50$ runs of PSO(40) is 56.29. The difference $\overline{d} = \overline{y}_1 - \overline{y}_2$ is 51.73. Since this value does not exceed 74.44, RU does not reject H.* ■

Example 1.2 shows a typical application of the Neyman-Pearson theory of testing. NPT has been under attack for many years. We will discuss important objections against NPT in the following and present an approach (NPT*) developed by Mayo to avoid these difficulties.

1.6.3 The Objectivity of NPT: Attacks and Misunderstandings

The specification of a significance level α or error of the first kind is a crucial issue in statistical hypothesis testing.

Hence, it is not surprising that attacks on the objectivity of NPT start with denying the objectivity of the specification of α. Why do many textbooks recommend a value of 5%?

Statement 3
"In no case can the appropriate significance level be determined in an objective manner" (Rubin, 1971).

Significance tests are tests of a null hypothesis. The significance level is often called the p-value. Another question that attacks the Neyman-Pearson theory of statistical testing refers to the p-value.

Statement 4
In the context of a statistical test, does a given p-value convey stronger evidence about the null hypothesis in a larger trial than in a smaller trial, or vice versa? (Gregoire, 2001).

Prima facie, one would answer that the p-value in a larger trial conveys stronger evidence. But the chance of detecting a difference increases as the sample size grows.
Although whole books devoted to these questions have been written, for example "The Significance Tests Controversy" (Morrison and Henkel, 1970), this controversy has not been recognized outside the statistical community. Gigerenzer states that there are no textbooks (written by and addressed to non-statisticians like psychologists) that explain differences in opinion about the logic of inference: "Instead, readers are presented with an intellectually incoherent mix of Fisherian and Neyman-Pearsonian ideas, but a mix presented as a seamless, uncontroversial whole: the logic of scientific inference" (Gigerenzer, 2003).
The following section discusses the definition of the p-value to clarify these questions.

1.6.4 The Objectivity of NPT: Defense and Understanding

Significance and the p-Value

Sometimes scientists claim that their results are significant because of the small p-value. The p-value is taken as an indicator that the null hypothesis is true (or false). This is as wrong as claiming that the movement of the leaves in the trees causes windstorms in autumn. The p-value is

$$p = Pr\{ \text{ result from test-statistic, or greater } \mid \text{ null model is true } \},$$

and not a measure of

$$p = Pr\{ \text{ null model is true } \mid \text{ test-statistic } \}.$$

Therefore, the p-value has "no information to impart about the verity of the null model itself"(Gregoire, 2001). The p-value is not related to any probability whether the null hypothesis is true or false. J. Neyman and E.S. Pearson[2] proposed a framework of acceptance and rejection of statistical hypothesis instead of a framework of significance. A significant result is a beginning, not an end. "Eating beans and peas 'significantly' decreases the probability of getting lung cancer. But why on Earth?"(Hacking, 2001). The specification of a p-value depends on the context in which the need for an experimental test arises. Researchers can judge the possible consequences of a wrong decision. Mayo comments on Statement 3:

Statement 5
The fact that the same data leads to different conclusions depending on the specification of α is entirely appropriate when such specifications are intended to reflect the researcher's assessment of the consequences of erroneous conclusions (Mayo, 1983, p.315).

The specifications, which are made in using NPT, are what allows tests to avoid being sterile formal exercises.

The misconception is that NPT functions which test scientific claims directly, cf. arrow (1) in Figure 1.2, and functions from which the statistical hypotheses are derived, arrow (2), are collapsed. It is necessary to leave some room between the statistical and the scientific

[2] Egon Pearson should not be confused with his father Karl, who proposed a different philosophy.

conclusion. Statistically significant conclusions are not automatically scientifically meaningful (Cohen, 1995).

How can statistical tests in a scientific inquiry provide "means of learning"? Being able to use the distribution of the test statistic T, error probabilities can be objectively controlled. This provides objective scientific knowledge. Detecting discrepancies between the correct and the hypothesized models enables learning about phenomena, for example "Is the actual value of δ positively discrepant from the hypothesized value δ_0?"

Severity

From the distribution of the experimental test statistic T is known that it is possible for an observed $\overline{d}$ to differ from a hypothesis δ_0, when no discrepancy exists (between δ, the true difference in means, and δ_0) or vice versa. A suitable choice of T enables the variation of the size of the observed differences, in a known manner, with the size of the underlying discrepancies. A test can be specified so that it will not very often classify an observed difference as significant (and hence reject H) when no discrepancy of scientific importance is detected, and not very often fail to do so (and so accept H) when δ is importantly discrepant from δ_0. This principle can be expressed in terms of the *severity requirement* (SR). It leads to NPT*, an extension of NPT.

(SR) An experimental result e constitutes evidence in support of a hypothesis H just to the extent that:

- **(SR-1)** e fits H, and
- **(SR-2)** H passes a severe test with e.

The requirement (SR-2) can be further specified by means of the *severity criterion*: A hypothesis H passes a severe test T with outcome e just in case the probability of H passing T with an outcome such as e (i.e., one that fits H as well as e does), given H is false, is very low. Staley (2002) additionally formulates a *severity question* (SQ) as follows: The severity question needs to be addressed to determine whether a given experimental outcome e is evidence for a given hypothesis H:

(SQ) How often would a result like this occur, assuming that the hypothesis is false?

Note that the severity requirements are not related to the error of the second kind (β error). Based on the severity requirements we attempt to show how NPT* can be used to objectively interpret statistical conclusions from experimental tests.

An Objective Interpretation of Rejecting a Hypothesis

The question "Are the differences real or due to the experimental error?" is central for the following considerations. The meta-statistical evaluation of the test results tries to determine whether the scientific import is misconstrued. A positive difference between the true and the observed value of less than one or two standard deviation units is quite often caused by experimental error. Thus, small differences may often erroneously be confused with effects due to real differences. This problem can be tackled by selecting a small α error, because only observed differences as large as $z_\alpha \sigma_{\overline{d}}$ are taken to reject H (see Figure 1.3). But choosing a small α value alone is not sufficient, because the standard error $\sigma_{\overline{d}}$ depends on the sample

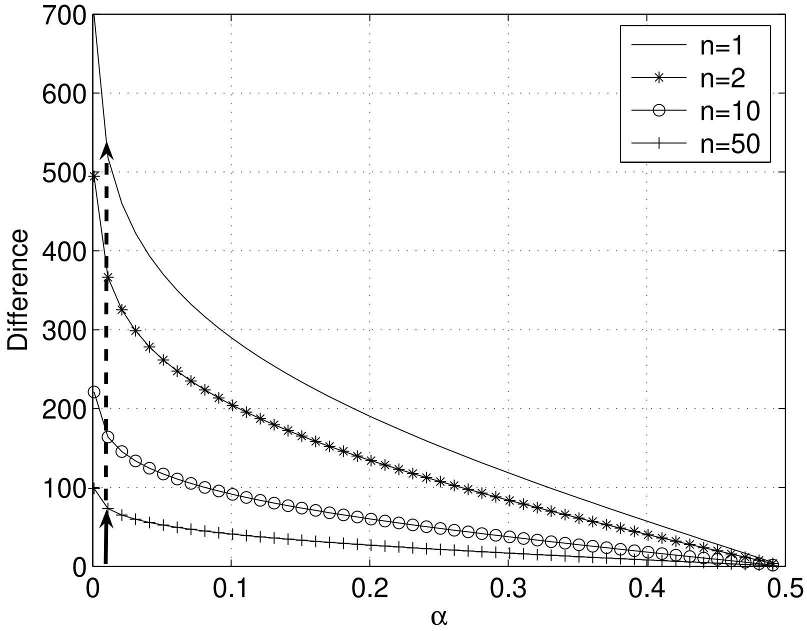


Figure 1.3: *Influence of the sample size n on the test result. Plot of the difference* $z_\alpha \cdot \sigma_{\overline{d}}$ *versus* α*, with* $\sigma = 160$*. The arrows point out the influence of the sample size for a given error of the first kind (*$\alpha = 0.01$*) on the difference: If n=1 the difference* $\overline{d} = \overline{Y}_1 - \overline{Y}_2$ *must exceed* 526.39 *to reject the null hypothesis. If more experiments are performed (*$n = 50$*), this difference must exceed only the seventh part of this value:* 74.44*. To demonstrate that there is no difference in means, experimenters can reduce the sample size n. Otherwise, increasing the sample size n may lead to a rejection of the null hypothesis.*

size n. A *misconstrual* is a wrong interpretation resulting from putting a wrong construction on words or actions. Mayo (1983) describes the first misconstrual (MC) as follows:

(MC-1) A test can be specified that will almost give rise to an $\overline{d} = \overline{y_1} - \overline{y_2}$ that exceeds δ_0 by the required difference $z_\alpha \sigma_{\overline{d}}$, even if the underlying δ exceeds δ_0 by as little as one likes.

This can be accomplished by selecting an appropriately large sample size n. If one is allowed to go on sampling long enough ($n \to \infty$), then even if the null hypothesis H is true, one is assured of achieving a statistically significant difference from H. The likelihood that we can detect a difference (power) in the test increases.

Example 1.3 (Sample size)
Consider $Y_1 \sim \mathcal{N}(100, 5)$ *and* $Y_2 \sim \mathcal{N}(110, 5)$*. Choosing a test with* $n = 50$ *will almost give rise to a* $\overline{Y}_{21}$ *that exceeds* δ_0 *by the required difference. Note that we wish to reject H if one mean is larger than the other. Therefore we are interested in the difference* $\overline{Y}_{21} = \overline{Y}_2 - \overline{Y}_1$*. Next, consider* Y_1 *and* $Y_3 \sim \mathcal{N}(100.1, 5)$*. If the sample size is increased, say* $n = 5000$*, a similar result can be obtained.* ■

Summarizing, the product $z_\alpha \sigma_{\overline{d}}$ can be modified by changing the sample size n or the error α:

$$\lim_{\alpha \to 0} z_\alpha = \infty \tag{1.6}$$

$$\lim_{n \to \infty} \sigma_{\overline{d}} = 0 \tag{1.7}$$

NPT allows misconstruals of the scientific import if a rejection of H is automatically taken to indicate that the scientific claim $\mathcal{C}$ is true. Even scientifically unimportant differences are classified as important because the test is too sensitive or powerful. When A and B are different treatments with associated means μ_A and μ_B, μ_A and μ_B are certain to differ in some decimal place so that $\mu_A - \mu_B = 0$ is known in advance to be false (Cohen, 1990; Tukey, 1991).

The Observed Significance Level

The frequency relation between a rejection of the null hypothesis and values of the difference in means, δ, is important for the interpretation of the rejection. To interpret the rejection of H, Mayo introduces the *observed significance level* $\alpha_{\overline{d}}(\delta)$

$$\alpha_{\overline{d}}(\delta) = \alpha(\overline{d}, \delta) = Pr(\overline{Y_1} - \overline{Y_2} \geq \overline{d} | \delta). \tag{1.8}$$

Hence, $\alpha_{\overline{d}}(\delta)$ is the area under the normal curve to the right of the observed $\overline{d}$ as illustrated in Figure 1.4. If we set $\delta_0 = 0$, then $\alpha_{\overline{d}}(\delta_0)$ is the frequency of an error of the first kind. If $\alpha_{\overline{d}}(\delta_0) \leq$ "the preset significance level of the test *RU*", then *RU* rejects H with $\overline{d}$. Rejecting H with *RU* is a good indicator that $\delta > \delta_0$ to the extent that such a rejection is not typical when δ is as small as δ_0. If any and all positive discrepancies from δ_0 are regarded as scientifically important, then a small $\alpha_{\overline{d}}(\delta)$ value ensures that construing such a rejection as indicating a scientifically important δ does not occur very often. Small $\alpha_{\overline{d}}(\delta)$ value does not prevent a *RU* rejection of H from often being misconstrued when relating it to the scientific claim $\mathcal{C}$, if some δ values in excess of δ_0 are still not considered scientifically important.

Regard the values of $\alpha_{\overline{d}}(\delta')$ for $\delta' \in \Omega_J$. A *RU* rejection with $\overline{d}$ successfully indicates that $\delta > \delta'$ if $\alpha_{\overline{d}}(\delta')$ is small. If $\alpha_{\overline{d}}(\delta')$ is fairly large, then such a rejection is the sort of event that fairly frequently occurs when $\delta \leq \delta'$.

To relate the statistical result to the scientific import, Mayo proposes to define δ_{un}

$$\delta_{\text{un}} = \text{ the largest scientifically unimportant value in excess of } \delta_0. \tag{1.9}$$

If $\alpha_{\overline{d}}(\delta_{\text{un}})$ is large, then the statistical result is not a good indication that the scientific claim is true. In addition to δ_{un}, we can define δ^α, the *inversion of the observed significance level* function as

$$\delta^\alpha = \text{ the value of } \delta \text{ in } \Omega \text{ for which } \alpha_{\overline{d}}(\delta) = \alpha. \tag{1.10}$$

Example 1.4
Consider a sample size of $n = 50$. *If* $\delta_{un} = 30$, *then rejecting H with RU cannot be taken as an indication that the scientific claim "PSO(40) outperforms PSO(20)" is true. The arrow in Figure 1.5 illustrates this situation. The observed significance level* $\alpha_{\overline{d}}(30) = 0.25$ *is not a strong indication that* δ *exceeds* 30. *However, if the sample size is increased* ($n = 500$), *then* $\alpha_{\overline{d}}(30) = 0.05$ *is small.*

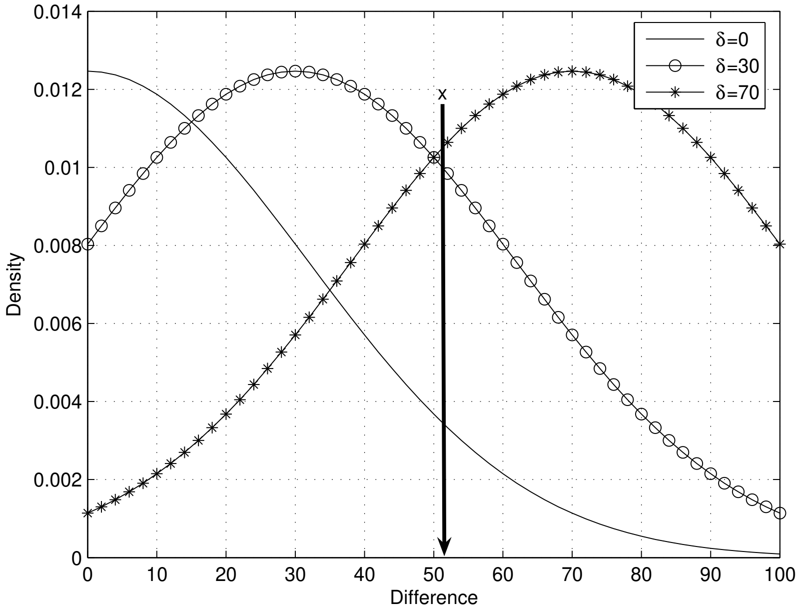


Figure 1.4: *Observed difference and three hypothetical differences. Difference in means for $n = 50$ samples and standard deviation $\sigma = 160$. The value from the test statistic $\overline{d} = 51.73$ remains fixed for varying means δ_i of different distributions associated with the null hypotheses H_i, $i = 1, 2, 3$. The figure depicts the probability density functions of the associated normal distributions for three different means: $\delta_1 = 0$, $\delta_2 = 30$, and $\delta_3 = 70$. To interpret the results, consider a hypothetical difference in means of $\delta_2 = 30$: The observed significance level $\alpha_{\overline{d}}(\delta_2)$ is the area under the normal curve to the right of $\overline{d}$. The value $\alpha_{51.75}(30) = 0.25$ is quite large and therefore not a good indication that the true difference in means is as large as $\delta_2 = 30$.*

Consider Example 1.1 and an observed significance level $\alpha = 0.5$. Then the value of the inversion of the observed significance level function is $\delta^{0.5} = 51.73$. As $\delta^{0.027} = \overline{y} - 2\sigma_{\overline{d}} = 31.49$ ($n = 500$), hence a RU rejection is an indication of $\delta > \delta^{0.027}$. ■

But are these results good indications that one is observing a difference $\delta > 0$, that is also scientifically important? This problem is outside the domain of statistics. Its answer requires the specification of a scientifically important difference, a reasonable sample size, and an acceptable error of the first kind, cf. Statement 5. The $\alpha_{\overline{d}}(\delta)$ function provides a non-subjective tool for understanding the δ values, a meta-statistical rule that enables learning on the basis of a given *RU* rejection. As the examples demonstrate, NPT* tools enable the experimenter to control error probabilities in an objective manner. The situation considered so far is depicted in Figure 1.5.

An Objective Interpretation of Accepting a Hypothesis

In a similar manner as rejecting H with a test that is too sensitive may indicate scientifically important δ's have been found, accepting a hypothesis with a test that is too insensitive may fail to indicate that no important δ's have been found. This can be defined as the second misconstrual:

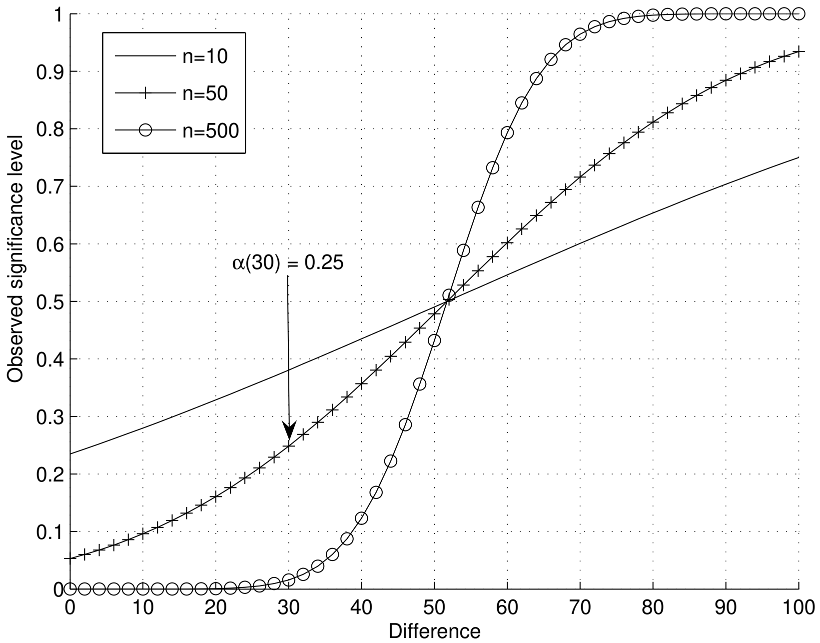


Figure 1.5: *Plot of the observed significance level $\alpha_{\overline{d}}(\delta)$ as a function of δ, the possible true difference in means. Lower $\alpha_{\overline{d}}(\delta)$ values support the assumption that there is a difference as large as δ. The measured difference is $\overline{d} = 51.73$, the standard deviation is $\sigma = 160$, cf. Example 1.1. The arrow points to the associated value of area under the normal curve to the right of the observed difference $\overline{d}$ as shown in Figure 1.4. Each point of the three curves shown here represents one single curve from Figure 1.4. The values can be interpreted as follows: Regard $n = 50$. If the true difference is a) 0, b) 51.73, or c) 100, then a) $H : \delta = 0$, b) $H : \delta = 51.73$, or c) $H : \delta = 100$ is wrongly rejected a) 5%, b) 50%, or c) 95% of the time.*

(MC-2) A test can be specified that will almost give rise to an $\overline{d}$ that does not exceed δ_0 by the required difference $z_\alpha \sigma_{\overline{d}}$, even if the underlying δ exceeds δ_0 by as much as one likes.

To interpret the acceptance of H with RU, Mayo defines

$$\beta_{\overline{d}}(\delta) = \beta(\overline{d}, \delta) = Pr(\overline{Y_1} - \overline{Y_2} \leq \overline{d} | \delta), \tag{1.11}$$

$$\delta^\beta = \text{ the value of } \delta \text{ in the parameter space } \Omega \text{ for which } \beta_{\overline{d}}(\delta) = \beta, \tag{1.12}$$

and

$$\delta_{\text{im}} = \text{ the smallest scientifically important } \delta \text{ in excess of } \delta_0. \tag{1.13}$$

Learning

NPT* accomplishes the task of objectively interpreting statistical results. The testing rule RU requires assumptions on the distribution of the underlying empirical observations $\mathcal{O}$. This is seen as part of task (3), depicted as arrow (3) in Figure 1.2. For example, one has to verify that the sample observation $\mathcal{O}$ can be modeled as the result of n independent observations of a random variable distributed according to the probability model $M(\delta)$. The assumption of independence can be checked using various goodness-of-fit tests. The learning function of tests may be accomplished even if the test assumptions are not satisfied precisely. NPT methods are robust and NPT* makes this robustness explicit.

To present a comprehensive example, we assumed a population that follows a gaussian distribution with known variance. Based on the bootstrap, that will be detailed in Section 2.3, we are able to use our approach independently from any assumptions on the underlying distribution.

Example 1.5 (Rejecting a hypothesis)
Consider the situation depicted in Figure 1.6. The experimental test statistic $T = \overline{Y_1} - \overline{Y_2}$ *is based on samples* $Y_1 \sim \mathcal{N}(110, 5)$ *and* $Y_2 \sim \mathcal{N}(100, 5)$*. The plot of the observed significance (Figure 1.6 on the left), indicates that one is observing a difference* $\delta > 0$*, and that this difference is not due to an increased sample size* n *alone. The values from Table 1.1 reflect this assumption: The observed significance levels for* 1 *and* 2 *standard error units,* $\alpha_{\overline{d}}(\sigma_{\overline{d}})$ *and* $\alpha_{\overline{d}}(2\sigma_{\overline{d}})$*, are small. This case will be referred to as RE-1 in the remainder of this thesis.* ■

Example 1.6 (Accepting a hypothesis)
The curves of the observed significance level $\alpha_{\overline{d}}$ *change their shape significantly if the true difference in means is very small, i.e. if* $Y_1 \sim \mathcal{N}(100.1, 5)$ *and* $Y_2 \sim \mathcal{N}(100, 5)$*. Figure 1.7 depicts this situation. Only a large sample size, i.e.* $n = 5000$*, is able to detect this difference, smaller sample sizes, i.e.* $n = 10$*, or* $n = 50$*, do not indicate a difference in means. Note, that in addition to the absolute* $\alpha_{\overline{d}}$*-values, the slope (the rate of change) is of importance. Figure 1.7 gives no reason to reject the null hypothesis. This case will be referred to as RE-2 in the remainder of this thesis.*
The corresponding plot that gives reason to accept the null hypothesis is shown in Figure 1.8. Consider a sample size of $n = 5000$*: The corresponding curve shows that we can safely state "there is no difference in means as large as* $\delta = 0.4$*." Figure 1.8 (right) shows a similar situation with reduced noise levels (*$Y_1 \sim \mathcal{N}(100.1, 1)$ *and* $Y_2 \sim \mathcal{N}(100, 1)$*). Regarding* $n = 5000$*, we can safely state that "there is no difference in means as large as* $\delta = 0.2$*." This case will be referred to as AC-1 in the remainder of this thesis.* ■

After discussing the objective interpretation of accepting or rejecting a hypothesis it is important to note that experiments consists of several tests. We have described one basic procedure only. In general, the learning process requires numerous statistical tests, the problem is broken up into smaller pieces. "One is led to break things down if one wants to learn" (Mayo, 1997, p. 254).

1.6.5 Related Approaches

Selvin (1970, p. 100) states that there is a difficulty in the interpretation of "significance" and "level of significance". The level of significance is the probability of wrongly rejecting the null hypothesis that there is no difference between two populations. The significance describes the scientific meaning of this difference.

In addition to this, the observed significance level is closely related to *consonance intervals* as introduced in Kempthorne and Folks (1971). Consonance intervals can be regarded as an inversion of significance tests: We ask for the degree of agreement of the parameters of a particular model with the data (Folks, 1981; Kempthorne and Folks, 1971). Given the data,

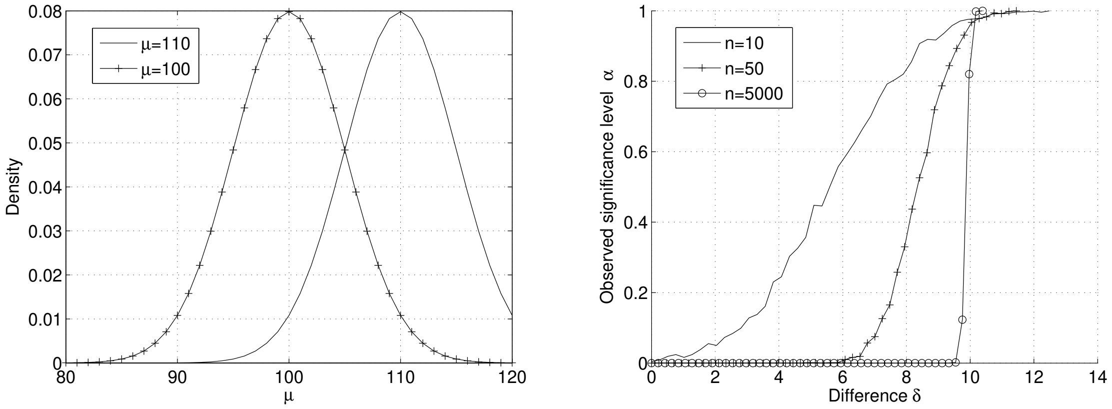


Figure 1.6: *Case RE-1. Rejecting a hypothesis. Density and observed significance level plots. $Y_1 \sim \mathcal{N}(110, 5)$, $Y_2 \sim \mathcal{N}(100, 5)$. This difference is meaningful and should be detected. These cases correspond to the configurations R1-R3 from Table 1.1.*

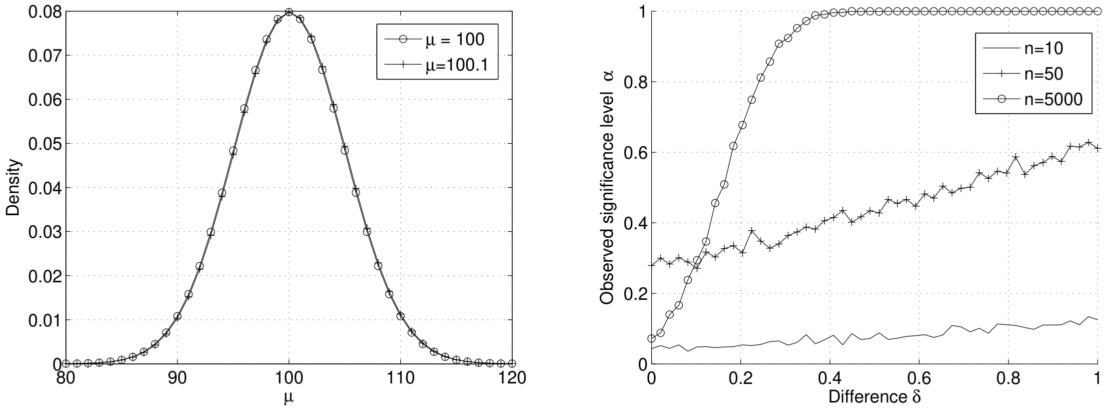


Figure 1.7: *Case RE-2. Rejecting a hypothesis. Density and observed significance level plots. $Y_1 \sim \mathcal{N}(100.1, 5)$, $Y_2 \sim \mathcal{N}(100, 5)$. This difference is not meaningful.These cases correspond to the configurations R4-R6 from Table 1.1.*

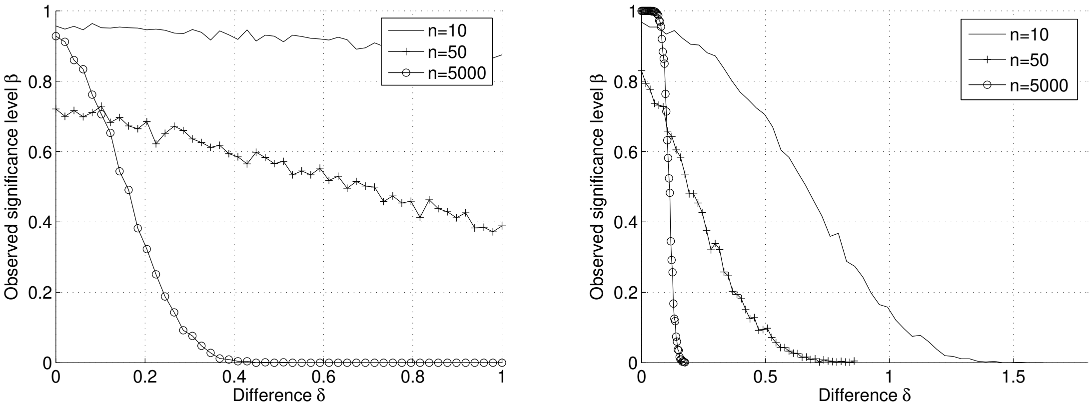


Figure 1.8: *Case AC-1. Accepting a hypothesis. Observed significance level plots. Left:* $Y_1 \sim \mathcal{N}(100.1, 5)$, $Y_2 \sim \mathcal{N}(100, 5)$. *Probably, the differences are not meaningful, because only large sample sizes produce small p-values. Right:* $Y_1 \sim \mathcal{N}(100.1, 1)$, $Y_2 \sim \mathcal{N}(100, 1)$. *These cases correspond to the configurations A1-A6 from Table 1.1.*

Table 1.1: *Rejection and acceptance of hypotheses. The sample size, n, the standard error,* $\sigma_{\overline{d}}$, *the observed difference,* $\overline{d}$, *the observed difference minus 1 and 2* $\sigma_{\overline{d}}$*'s, and the values of the observed significance levels for 1 and 2* $\sigma_{\overline{d}}$*'s are shown.*

Conf.	n	$\sigma_{\overline{d}}$	$\overline{d}$	$\overline{d} - \sigma_{\overline{d}}$	$\overline{d} - 2\sigma_{\overline{d}}$	$\alpha_{\overline{d}}(\sigma_{\overline{d}})$	$\alpha_{\overline{d}}(2\sigma_{\overline{d}})$
$R1$	10	1.99	9.58	7.59	5.60	2.69×10^{-4}	0
$R2$	50	0.94	10.66	9.72	8.79	0	0
$R3$	5000	0.1	9.89	9.79	9.69	0	0
$R4$	10	1.57	2.99	1.41	−0.16		
$R5$	50	1.06	0.7	−0.36	−1.43		
$R6$	5000	0.1	0.18	0.08	−0.02	0.29	0.67
Conf.	n	$\sigma_{\overline{d}}$	$\overline{d}$	$\overline{d} + \sigma_{\overline{d}}$	$\overline{d} + 2\sigma_{\overline{d}}$	$\alpha_{\overline{d}}(\sigma_{\overline{d}})$	$\alpha_{\overline{d}}(2\sigma_{\overline{d}})$
$A1$	10	1.57	2.99	4.56	6.13		
$A2$	50	1.06	0.7	1.76	2.83		
$A3$	5000	0.1	0.18	0.28	0.38	0.71	0.33
$A4$	10	0.31	0.68	0.99	1.31	0.15	0.46
$A5$	50	0.21	0.22	0.43	0.65	0.48	0.14
$A6$	5000	0.02	0.12	0.14	0.16	1	1

the parameters of the model are evaluated. I.e. observing 503 heads in 1000 coin tosses, the model "heads and tails are equally probable" is consonant with the observation.

1.7 Popper and the New Experimentalists

In a similar manner as Gigerenzer (2003) presents his tools to theories approach, Mayo suspects that Popper's falsification theory is well accepted by many scientists since it reflects the standard hypothesis testing principles of their daily practice. To clarify the difference between Mayo's NPT* approach and popperian testing, the reader may consider the following quotation:

> Mere supporting instances are as a rule too cheap to be worth having: they can always be had for the asking; thus they cannot carry any weight; and any support capable of carrying weight can only rest upon ingenious tests, undertaken with the aim of refuting our hypothesis, if it can be refuted (Popper, 1983).

Popper's focus lies on the rejection of hypotheses, his concept of severity does not include tools to support the acceptance of scientific hypotheses as introduced in Section 1.6. Popper states that the theoretician will try to detect any false theories, he will "try to construct severe tests and crucial test situations" (Popper, 1979). But Popper does not present objective interpretations (as Mayo does) for accepting and rejecting hypotheses.

Mayo explicitly states that it is important to distinguish popperian severity from hers. Also Popper stresses the importance of severe tests: Hypothesis H passes a severe test with experimental result e if all alternatives to H that have been tested entail not-e. But there are many not-yet-considered or not-yet-even-thought-of alternative hypotheses that also entail e. Why is this alternative objection not relevant for NPT*? Mayo comments:

> Because for H to pass a severe test in my sense it must have passed a severe test with high power at probing the ways H can err. And the test that alternative hypothesis H' failed need not be probative in the least so far as H's errors go. So long as two different hypotheses can err in different ways, different tests are needed to probe them severely (Mayo, 1997, p. 251).

Hypothesis H has passed a severe test if the hypothesis itself has been tested. It is not sufficient—as claimed in the popperian paradigm—that all alternative hypotheses to H failed and H was the only hypothesis that passed the test.

And What About Theory?

According to the new experimentalists, experiment can have a life of its own. An experiment is independent of large-scale theory. This is an obvious contradiction to the popperian view that theory precedes experiments and that there is no experiment without theory. While discussing the relationship between theory and experiment, Hacking comments on Popper's statement: "Theory dominates the experimental work from its initial planning to the finishing touches in the laboratory" (Popper, 1959) with a counterexample that mentions Humphry Davy (1778–1829).

> Davy's noticing the bubble of air over the algae is one of these [obvious counterexamples]. It was not an 'interpretation in the light of theory' for Davy had

> initially no theory. Nor was seeing the taper flare an interpretation. Perhaps if he went on to say, 'Ah, then it is oxygen', he would have been making an interpretation. He did not do that. (Hacking, 1983)

We note additionally that a great many examples from the history of science can be mentioned as counterexamples to the popperian view that theory dominates the experimental work. Davy experimented with gases by inhaling them and thus invented the laughing gas, nitrous oxide, without any theory. And, in opposition to existing theories, Davy showed that hydrochloric acid did not contain oxygen. One last example from Davy: Ice cubes melt when they are rubbed together—in contradiction to the caloric theory. Summarizing the discussion about theory from this chapter, we conclude: Theory can be described as wishful thinking. Moreover it is defined though consulting a dictionary as:

1. The analysis of a set of facts in their relation to one another.
2. An abstract thought: speculation (Merriam-Webster Online Dictionary, 2004b).

The latter definition is consonant with Hacking's suggestion not to differentiate only between theory and experiment, but to use a tripartite division instead: Speculation, calculation and experimentation (Hacking, 1996). We will not enforce the gap between theory and practice any further, because we sympathize with the idea of this tripartite division that will be reconsidered in Chapter 8. The reader is also referred to Hacking's "Representing and Intervening" that details these ideas.

1.8 Summarizing: Experiments

The results from this chapter can be summarized as follows:

1. The goal of this thesis is to prepare good grounds for experimental research in evolutionary computation.
2. Solely theoretical approaches to investigate, compare, and understand algorithms are not satisfactory from an experimenter's point of view.
3. Algorithm runs can be treated as experiments.
4. Guidelines from experimental algorithmics provide good starting points for experimental studies.
5. Experimental algorithmics is based on the popperian paradigms:
   (a) There is no experiment without high-level theories.
   (b) Theories should be falsifiable.
6. We claim that:
   (a) There are experiments without high-level theories ("experiment can have a life of its own").
   (b) Popper's falsification should be complemented with validation.

7. The concept of the new experimentalism is transferred from philosophy to computer science, especially to evolutionary computation.

8. Models are central elements of an understanding of science.

9. Mayo introduced models of statistical testing that leave room between scientific claims and statistical hypotheses.

10. Hypothesis testing can be applied as an automatic rule (NPT) or as a learning tool (NPT*).

11. The approach presented here enables learning from experiments. Learning can be guided by plots of the observed significance against the difference as shown in Figure 1.5.

Example 1.2 was based on the assumption of known variances and normally distributed data. The following section introduces statistical tools that enable the application of NPT* methods even if the underlying distribution is unknown.

Chapter 2

Statistics

> Like dreams, statistics are a form of wish fulfillment.
>
> Jean Baudrillard

2.1 Introduction

This chapter discusses some methods from classical and modern statistics. Statistics can be defined as a branch of mathematics dealing with the collection, analysis, interpretation, and presentation of masses of data (Merriam-Webster Online Dictionary, 2004a). The term "computational statistics" subsumes computationally intensive methods. They comprise methods ranging from exploratory data analysis to Monte Carlo methods. Data should be enabled to "tell their story". Many methods from computational statistics do not require any assumptions on the underlying distribution. Computer based simulations facilitate the development of statistical theories: 50 out of 61 articles in the theory and methods section of the Journal of the American Statistical Association in 2002 used Monte Carlo simulations (Gentle et al., 2004a).

The accuracy and precision of data may be limited due to noise. How can deterministic systems like computers model this randomness? Stochastic computer experiments, as performed in evolutionary computation, have to cope with a different concept of randomness than agricultural or industrial experiments. The latter face inherent randomness, whereas the former require methods to generate randomness. This is accomplished by generating sequences of *pseudo-random numbers*.

A sequence of infinite length is random, if the quantity of information it contains is infinite too. If the sequence is finite, it is formally impossible to verify whether it is random or not. This results in the concept of pseudo-randomness: Statistical features of the sequence in question are tested, i.e. the equiprobability of all numbers (Knuth, 1981; Schneier, 1996). Following this rather pragmatic approach, randomness and pseudo-randomness will be treated equivalently throughout the rest of this work.

First some basic definitions from hypothesis testing are introduced. Next, a bootstrap method to generate significance plots as shown in Figure 1.5 is described. It provides an effective method to use the raw data without making any additional assumptions on the

underlying distribution. The bootstrap has been used to solve problems that would be too complicated for classical statistical techniques.
Then some useful tools for the analysis of computer experiments are presented. Common to all these methods is that they provide means to explain the variability in the data. Consider an optimization algorithm with exogenous strategy parameters A and B. Varying these parameters may cause a change in the algorithm's performance Y. The goal of all methods presented here is to find out:

1. Which factor has a relevant influence on the performance?

2. What are good factor settings that guarantee that the algorithm performs well?

Regression models provide means for a numerical analysis. Half-normal plots and interaction plots can complement these results graphically. Scatter plots and regression trees are distribution-free methods to visualize structure in data. This chapter concludes with an introduction of the basic definitions for design and analysis of computer experiments (DACE) models.

A comprehensive introduction into the statistical methods cannot be given here. This chapter attempts to focus on the basic ideas. Montgomery (2001) presents an introduction to the classical design of experiment, the bootstrap is introduced in Efron and Tibshirani (1993), classification and regression trees are discussed in Breiman et al. (1984), and Santner et al. (2003) describe the design and analysis of computer experiments. Further reading is mentioned in the text.

2.2 Hypothesis Testing

The following subsections introduce the basic definitions used for statistical hypothesis testing. Example 1.2 is reconsidered. Besides the three procedures presented here, many more test procedures exist. These tests can be classified according to known or unknown variances, equal or unequal sample sizes, and equal or unequal variances (Montgomery, 2001). The z-test is presented first, because it has been used in Example 1.2. In contrast to the z-test, where variances have to be known, in the t-test estimates of the variances are computed.

2.2.1 The Two-Sample z-Test

In Example 1.2 the performances of two algorithms, PSO(20) and PSO(40) respectively, were compared. The vector $y_i = (y_{i1}, \dots, y_{in_i})$ represents the n_i observations from the i-th algorithm, $\overline{y}_i$ denotes the i-th sample mean and σ_i^2 the associated variances. The distribution of the difference in means $\overline{Y}_{12} = \overline{Y}_1 - \overline{Y}_2$ is $\mathcal{N}(\overline{y}_1 - \overline{y}_2, \sigma^2(1/n_1 + 1/n_2))$, if the samples were drawn from independent normal distributions $\mathcal{N}_i(\mu_i, \sigma_i^2)$ with common variance $\sigma^2 = \sigma_i^2$, $i = 1, 2$. If σ^2 were known and

$$\mu_1 = \mu_2, \tag{2.1}$$

then

$$Z_0 = \frac{\overline{y}_1 - \overline{y}_2}{\sigma\sqrt{1/n_1 + 1/n_2}} \sim \mathcal{N}(0, 1).$$

Equation 2.1 is a statement or a statistical hypothesis about the parameters of a probability distribution, the null hypothesis: $H: \ \mu_1 = \mu_2$. The alternative hypothesis can be defined as the statement $J:\ \mu_1 \neq \mu_2$. The procedure of taking a random sample, computing a test statistic, and the accepting (or failing to accept) of the null hypothesis H is called *hypothesis testing*. The *critical region* contains the values that lead to a rejection of H. The one-sided alternative hypothesis can be specified as $J:\ \mu_1 > \mu_2$. The *significance level* α is the probability of a type I error for the test:

$$\alpha = Pr(\text{type I error}) = Pr(\text{reject } H|H \text{ is true}). \tag{2.2}$$

The type II error is defined as

$$\beta = Pr(\text{type II error}) = Pr(\text{fail to reject } H|H \text{ is false}). \tag{2.3}$$

To determine whether to reject the null hypothesis $H: \mu_1 = \mu_2$, the value of the test statistic $T: \overline{d} = \overline{y}_1 - \overline{y}_2$ is compared to the normal distribution. If

$$\overline{d} = \overline{y}_1 - \overline{y}_2 \geq z_\alpha \sigma \sqrt{1/n_1 + 1/n_2},$$

where z_α is the *upper α percentage point* of the normal distribution, the null hypothesis would not be accepted in the classical *two-sample test*. When $\alpha = 0.01$, then z_α has the value 2.23. With $n_1 = n_2 = n$ and $\sigma = 160$ follows that $z_\alpha \sigma \sqrt{1/n_1 + 1/n_2} = 2.23 \cdot 160\sqrt{2/50} = 74.44$ as in Example 1.2.

The definition of the upper α percentage point of the normal distribution can be generalized to the case of more than one random variable. Let $(W_1, \ldots, W_k)$ have the k-variate normal distribution with mean vector zero, unit variances, and common correlation ρ. Then

$$P\left(\max_{1 \leq i \leq k} W_i \leq Z_{k,\rho}^{(\alpha)}\right) = 1 - \alpha \tag{2.4}$$

defines the upper-α equicoordinate critical point $Z_{k,\rho}^{(\alpha)}$ of this distribution (Bechhofer et al., 1995, p. 18).

2.2.2 The Two-Sample t-Test

If the variances of the populations are unknown, the sample variances

$$S_i^2 = \frac{\sum_{k=1}^{n_i} (y_{ik} - \overline{y}_i)^2}{n_i - 1}$$

can be used to estimate σ_i^2, $i = 1, 2$. The related test procedure is called the *two-sample t-test*. The upper α percentage point of the normal distribution is replaced by t_{α, n_1+n_2-2}, the upper α percentage point of the t-distribution with $n_1 + n_2 - 2$ degrees of freedom.

Let $S_p^2 = ((n_1 - 1)S_1^2 + (n_2 - 1)S_2^2)/(n_1 + n_2 - 2)$, the pooled variance, denote an estimate of the common variance σ^2. Then

$$t_0 = \frac{\overline{y}_1 - \overline{y}_2}{S_p^2 \sqrt{1/n_1 + 1/n_2}}. \tag{2.5}$$

If H is true, t_0 is distributed as $t_{n_1+n_2-2}$, and $100(1-\alpha)$ percent of the values of t_0 lie in the interval $[-t_{\alpha/2, n_1+n_2-2}, t_{\alpha/2, n_1+n_2-2}]$, where $t_{\alpha,n}$ denotes the upper α percentage point of the t-distribution with n degrees of freedom.

The t-distribution with $n_1 + n_2 - 2$ degrees of freedom is called the relevance distribution for the test statistic t_0. To reject H only if one mean is larger than the other ($\mu_1 > \mu_2$), the criterion $t_0 > t_{\alpha,n_1+n_2-2}$ is used. This is the one-sided t-test.

From Equation 2.5 follows

$$Pr(-t_{\alpha/2,n_1+n_2-2} \leq \frac{\overline{y}_1 - \overline{y}_2 - (\mu_1 - \mu_2)}{S_p\sqrt{1/n_1 + 1/n_2}} \leq t_{\alpha/2,n_1+n_2-2}) = 1 - \alpha,$$

and therefore

$$\begin{aligned} \overline{y}_1 - \overline{y}_2 - t_{\alpha/2,n_1+n_2-2} S_p\sqrt{1/n_1 + 1/n_2} \\ \leq \mu_1 - \mu_2 \\ \leq \overline{y}_1 - \overline{y}_2 + t_{\alpha/2,n_1+n_2-2} S_p\sqrt{1/n_1 + 1/n_2} \end{aligned}$$

is a $100(1 - \alpha)$ percent confidence interval for $\mu_1 - \mu_2$.

2.2.3 The Paired t-Test

To compare different run configurations, variance reducing techniques such as common random numbers have been used in our experiments. The jth paired difference

$$d_j = y_{1j} - y_{2j} \qquad j = 1, \ldots, n,$$

is used to define the test statistic

$$t_0 = \frac{\overline{d}}{S_d/\sqrt{n}},$$

where $\overline{d} = \frac{1}{n}\sum_{j=1}^{n} d_j$, and

$$S_d = \sqrt{\sum_{j=1}^{n} \frac{(d_j - \overline{d})^2}{n-1}}, \tag{2.6}$$

is the *sample standard deviation of the differences.* The null hypothesis $H : \mu_1 = \mu_2$, or equivalently $H : \delta = 0$, would be not accepted if $t_0 > t_{\alpha,n-1}$. The paired t-test can be advantageous compared to the two-sample t-test due to its noise reduction properties. The confidence interval based on the paired test can be much narrower than the corresponding interval from the two-sample test. The reader is referred to the discussion in Montgomery (2001).

2.3 Monte Carlo Simulations

The statistical approach from Example 1.2 requires the following steps:

1. At first a sampling distribution for a statistic is derived.

2. Then the probability of a sample statistic is determined.

Many sampling distributions rely on statistical assumptions; consider for example the assumption that samples are drawn from normal distributions like for the t-distribution. Furthermore classical techniques often apply asymptotic results under the assumption that the size of the available set of samples is sufficiently large.

Monte Carlo simulations can be applied for known population distributions from which the samples are drawn and unknown sampling distributions of the test statistic, for example the trimmed mean or the interquartile range.

As bootstrap methods treat the sample as the population, they can be applied if the sampling distribution is unknown (Efron and Tibshirani, 1993). They require a representative sample of the population. Nowadays the bootstrap is considered a standard method in statistics (Mammen and Nandi, 2004). It has been successfully applied to solve problems that would be too complicated for classical statistical techniques and in situations where the classical techniques are not valid (Zoubir and Boashash, 1998).

Bootstrap

The idea behind the *bootstrap* is similar to a method that is often applied in practice. Experiments are repeated to improve the estimate of an unknown parameter. If a representative sample is available, the bootstrap randomly reassigns the observations, and recomputes the estimate. The bootstrap is a computational intensive technique. Let $\hat{\theta}$ be the estimate of an unknown parameter θ that has been determined by calculating a statistic T from the sample:

$$\hat{\theta} = T = t(y_1, \dots, y_n).$$

By sampling with replacement, n_b bootstrap samples can be obtained. The bootstrap replicates of $\hat{\theta}$

$$\hat{\theta}^{*b} = t(y^{*b}), \qquad b = 1, \dots, n_b$$

provide an estimate of the distribution of $\hat{\theta}$. The bootstrap procedure is described in Figure 2.1.

How can the bootstrap be used to generate plots of the observed significance $\alpha_{\overline{d}}(\delta)$? Let $y_1 = (y_{11}, \dots, y_{1n})^T$ and $y_2 = (y_{21}, \dots, y_{2n})^T$ denote the random samples, and $d = y_1 - y_2 = (y_{11} - y_{21}, \dots, y_{1n} - y_{2n})^T$ their difference vector. The procedure to obtain an estimate of the observed significance level $\alpha_{\overline{d}}(\delta)$ for a difference $\delta = d_0$ under the null hypothesis H can be implemented as shown in Figure 2.2.

Example 2.1 (Bootstrap)
Let y_1 and y_2 denote two vectors with representative samples from a population. If $a \in \mathbb{R}$ and the vector $y = (y_1, \dots, y_n)^T \in \mathbb{R}^n$, the scalar-vector addition is defined as

$$a + y = (y_1 + a, \dots, y_n + a)^T.$$

The bootstrap approach that has been used to generate the plots of the observed significance comprises the steps shown in Figure 2.2. They can be detailed as follows:

1. *Determine $d = y_1 - y_2$.*
2. *Determine $\overline{d} = 1/n \sum_{j=1}^{n} (y_{1j} - y_{2j})$.*
3. *Determine $d_0 = d - \overline{d}$.*
4. *Specify the lower bound d_l and the upper bound d_u for the plot.*

Algorithm 2.1 (Generic Bootstrap)

1. *Calculate $\hat{\theta}$ from a representative sample $y = (y_1 \dots, y_n)$.*
2. *To generate the bootstrap data sets $y^{*b} = (y_1^{*b}, \dots, y_n^{*b})$ sample with replacement from the original sample.*
3. *Use the bootstrap sample y^{*b} to determine $\hat{\theta}^{*b}$.*
4. *Repeat steps 2 and 3 n_b times.*
5. *Use this estimate of the distribution of $\hat{\theta}$ to obtain the desired parameter, for example the mean.*

Figure 2.1: *The generic bootstrap procedure.*

Algorithm 2.2 (Bootstrap for the observed significance)

1. *Determine the mean $\overline{d}$ from the two samples y_1 and y_2.*
2. *Generate n_b bootstrap sample sets d^{*b}, $b = 1, \dots, n_b$ from $d_0 = d - \overline{d}$.*
3. *Determine the n_b mean values $\overline{d}^{*b}$.*
4. *Determine n, that is the number of times that $\overline{d}^{*b} > \overline{d}$.*
5. *The ratio $r = n/n_b$ gives the desired value.*

Figure 2.2: *Basic bootstrap procedure to determine the observed significance level $\alpha_{\overline{d}}(\delta)$. It can be applied to generate plots of the observed significance as shown in Figure 1.5. It requires only two paired and representative samples y_1 and y_2.*

5. *Specify m, the number of points to be plotted in the interval $[d_l, d_u]$.*

6. *For $i = 1$ to m do:*

   (a) *Determine $d_i = d_l + i \cdot (d_u - d_l)/m + d_0$.*

   (b) *Generate n_b bootstrap sample sets d_i^{*b}, $b = 1, \dots, n_b$ from d_i.*

   (c) *Determine the n_b mean values $\overline{d}_i^{*b}$.*

   (d) *Determine n_i, that is the number of times that $\overline{d}_i^{*b} > \overline{d}$.*

   (e) *Determine the ratio $r_i = n_i/n_b$.*

Finally, the m points $(d_0^{(i)}, r^{(i)})$ are plotted. The ratio r_i corresponds to the observed significance value $\alpha_{\overline{d}}(d_0^{(i)})$. ■

Histograms of the bootstrap replicates as shown in Figure 2.3 are appropriate tools for examining the distribution of $\hat{\theta}$. Figure 2.4 depicts the result based on the bootstrap. It represents the same situation as shown in Figure 1.5, without making any assumption on the underlying distribution. As the sample size is increased, i.e. from 50 to 500, the bootstrap and the true curve start to look increasingly similar.

The following section presents standard definitions used in classical and modern statistics, especially in the design and analysis of experiments.

2.4 DOE: Standard Definitions

The *design of experiments* (DOE) has a long tradition in statistics. It has been developed for different areas of applications: Agriculture (Fisher, 1935), industrial optimization (Box et al., 1978), computer simulation (Kleijnen, 1987), and deterministic computer experiments (Sacks et al., 1989).

The following definitions are commonly used in DOE. The input parameters and structural assumptions to be varied during the experiment are called *factors* or *design variables*. Other names frequently used are predictor variables, input variables, regressors, or independent variables. The vector of design variables is represented as $x = (x_1, \dots, x_k)^T$. Different values of parameters are called *levels*. The levels can be scaled to the range from -1 to $+1$. Levels can be qualitative, i.e. selection scheme, or quantitative, i.e. population size. The *design space*, the region of interest, or the experimental region is the k-dimensional space defined by the lower and upper bounds of each design variable. A *sample* or a *design point* is a specific instance of the vector of design variables. An *experimental design* is a procedure for choosing 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”(Kleijnen, 2001). One parameter design setting is run for different pseudo-random number settings, resulting in replicated outputs. The output value y is called *response*, other names frequently used are output variables or dependent variables.

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.

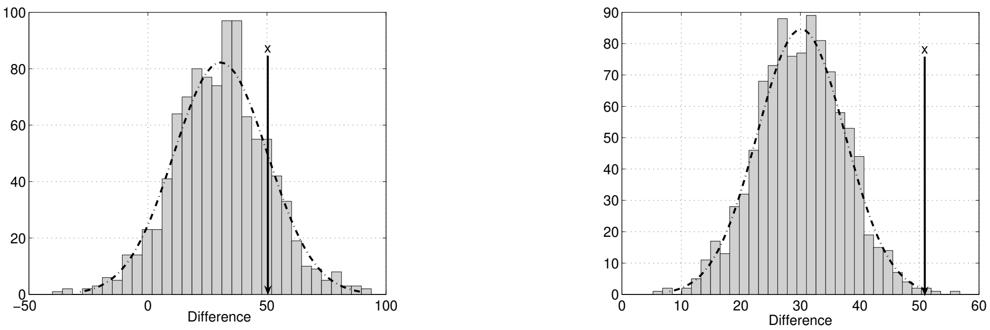


Figure 2.3: *Histograms of the bootstrap samples. Left: 50 repeats, right: 500 samples. These figures show histograms of the bootstrap samples that were generated at step 6b in Example 2.1. The difference* d_i *has the value* 30. *The dash dotted curves show the superimposed normal density. The area to the right of* $\overline{d} = 51.73$ *under the curve corresponds approximately with the observed significance level* $\alpha_{\overline{d}}(\delta)$, *the ratio* r_i.

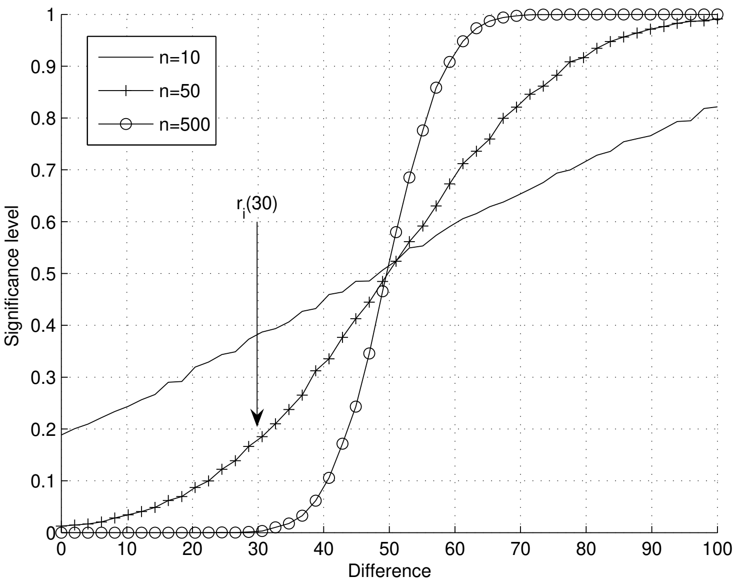


Figure 2.4: *This figure depicts the same situation as shown in Figure 1.5. But unlike in Figure 1.5, no assumptions on the underlying distribution have been made. Samples of size* $n = 10$, 50, *and* 500 *respectively have been drawn from a normal distribution. The bootstrap procedure described in Example 2.1 has been used to generate this plot. The curves look qualitatively similar to the curves from Figure 1.5. As the number of samples increases, the differences between the exact and the bootstrap curves becomes smaller. The measured difference is* 51.73, $\sigma = 160$, *cf. Example 1.1. Regard* $n = 50$*: If the true difference is a) 0, b) 51.73, or c) 100, then a)* $H : \delta = 0$, *b)* $H : \delta = 51.73$, *or c)* $H : \delta = 100$ *is (approximately) wrongly rejected a) 1%, b) 50%, or c) 99% of the time.*

2.5 The Analysis of Variance

One of the most useful principles in inferential statistics, the (single) factor *analysis of variance* (ANOVA) is introduced next (Montgomery, 2001). First, the dot subscript notation is defined: Consider m different treatments. The sum of all observations under the ith treatment is

$$y_{i\cdot} = \sum_{j=1}^{n} y_{ij}$$

Then, $\overline{y}_{i\cdot} = y_{i\cdot}/n$, $i = 1, 2, \ldots, m$, and

$$y_{\cdot\cdot} = \sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij}, \qquad \overline{y}_{\cdot\cdot} = y_{\cdot\cdot}/N,$$

where $N = nm$ is the total number of observations. The *total corrected sum of squares*

$$SS_{\mathrm{T}} = \sum_{i=1}^{m}\sum_{j=1}^{n}(y_{ij} - \overline{y}_{\cdot\cdot})^2 \tag{2.7}$$

measures the total variability in the data. It can be partitioned into a sum of squares of the difference between the treatment averages and the grand average $\mathrm{SS}_{\mathrm{TREAT}}$ plus a sum of squares of the differences of observations within treatments from the treatment average SS_{E}:

$$\sum_{i=1}^{m}\sum_{j=1}^{n}(y_{ij} - \overline{y}_{\cdot\cdot})^2 = n\sum_{i=1}^{m}(\overline{y}_{i\cdot} - \overline{y}_{\cdot\cdot})^2 + \sum_{i=1}^{m}\sum_{j=1}^{n}(y_{ij} - \overline{y}_{i\cdot})^2 \tag{2.8}$$

This *fundamental ANOVA principle* can be written symbolically as:

$$SS_{\mathrm{T}} = SS_{\mathrm{TREAT}} + SS_{\mathrm{E}} \tag{2.9}$$

The term SS_{TREAT} is called the *sum of squares due to the treatments*, and SS_{E} is called the *sum of squares due to error.*

2.6 Linear Regression Models

(Linear) regression models are central elements of the classical design of experiments approach. In simulation and stochastic optimization, regression models can be represented as follows:

$$y = f_1(z_1, \ldots, z_k, r_0), \tag{2.10}$$

where f_1 is a mathematical function, e.g. $f_1 : \mathbb{R}^{k+1} \to \mathbb{R}$: Given the values of the argument z_i and at least one random number seed r_0, the simulation program or the optimization algorithm determine 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 or optimization data. Least square methods can be applied to estimate the linear model

$$y = X\beta + \epsilon, \tag{2.11}$$

where y denotes a column vector with the k responses, ϵ is the vector of k error terms, and β denotes the vector with q parameters β_j ($k \geq q$). Usually, the normality assumption (the error term ϵ is normal with *expectation* $E(\epsilon) = 0$ and *variance* $V(\epsilon) = \sigma^2$) is made. X is the $(k \times q)$ matrix of independent regression variables. The regression variables can represent the design variables from Section 2.4. The variable x_0 is the dummy variable equal to $\mathbf{1}$, the remaining $q - 1$ variables may correspond to the simulation parameters z_i. Therefore, we obtain the $(k \times q)$ *regression matrix*

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1,q-1} \\ \vdots & & & & \vdots \\ 1 & x_{i1} & x_{i2} & \cdots & x_{i,q-1} \\ \vdots & & & & \vdots \\ 1 & x_{k1} & x_{k2} & \cdots & x_{k,q-1} \end{pmatrix}. \tag{2.12}$$

Experimental settings (designs), where the regression matrix X satisfies $XX^T = kI$, are called orthogonal. The ordinary least squares (OLS) estimator of the vector of regression parameters β in Equation 2.11 reads:

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{2.13}$$

with covariance $cov(\hat{\beta}) = \sigma^2 (X^T X)^{-1}$. An estimator $\hat{\alpha}$ is unbiased, if $E(\hat{\alpha}) = \alpha$. If the errors in Equation 2.11 are i.i.d., then $\hat{\beta}$ is the *best linear unbiased estimator (BLUE)*.

An example how to apply regression models to analyze the performance of evolutionary algorithms is given in Bartz-Beielstein (2003).

Generalized Linear Models

Linear models are applicable to problems that have gaussian errors. 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 (McCullagh and Nelder, 1989). François and Lavergne (2001) and Bartz-Beielstein (2003) use generalized linear models to analyze evolutionary algorithms. Bartz-Beielstein et al. (2005b) propose GLMs to analyze and validate simulation models.

Logistic regression models that are based on the success ratio (SCR) can be used to analyze the algorithm's performance. Whether or not the optimization run has located a prespecified 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. 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 Collett (1991). Myers and Hancock (2001) present an example that uses a genetic algorithm to solve consistent labeling problems.

Standard textbooks on regression analysis such as Draper and Smith (1998) present methods of checking the fitted regression model. However, the fact that the regression model passes some test does not mean that it is the correct model. Graphical tools should be used to guide the analysis. Half-normal plots, scatter plots, interaction plots, and box plots that can be applied to analyze computer experiments will be presented next.

2.7 Graphical Tools

This section presents graphical tools that support the analysis of factor effects and interactions. Half-normal plots, interaction plots, and box plots can complement classical DOE methods. They are based on factorial designs (designs will be introduced in Chapter 4). Scatter plots can be used in combination with space filling designs. These designs are commonly used in modern design and analysis of computer experiments (DACE).

2.7.1 Half-Normal Plots

Half-normal plots are comprehensive tools for analyzing experimental data. They possess many features that can also be found in other statistical tools.

Least-squares estimation gives an estimate of the effect of a factor. The estimated effects minimize the sum of squared differences between raw data and the fitted values from the estimates. An ordered list of the main effects (and of the interactions as well) can be constructed. A half-normal plot is a plot of the absolute value of the effect estimates against their cumulative normal probabilities.

If a 2^k full or a fractional factorial design (these designs will be explained later on) was chosen to generate the experimental data, all least squares estimates for main effects and interactions have the following representation: $\hat{y} = \overline{y}_+ - \overline{y}_-$, where $\overline{y}_+$ and $\overline{y}_-$ denote the average value of all responses for which the factor was set to its high value or to its low value, respectively.

For large sample sizes these difference-of-sums for the estimated effects tend to follow a normal distribution. This follows from the central limit theorem under mild assumptions. Since all estimates present a difference of averages, their standard deviations will be the same (under the assumption of constant variances). Important factors will have a large difference, whereas unimportant factors are those that have near-zero effects.

If the experiment were such that no factors were important, then the estimated effects would behave like randomly drawn samples that follow a normal distribution and their normal probability plot is nearly linear. If some subset of factors were important, then these factors would be well off the line. As signs of the estimates have been chosen arbitrarily, the effect magnitudes can be considered and not the signed effects. This leads to half-normal plots that give a more compact presentation than normal plots.

Figure 2.5 depicts a typical half-normal plot that has been generated while optimizing an evolution strategy. Schwefel et al. (1995) and Beyer and Schwefel (2002) provide a comprehensive introduction to this special class of evolutionary algorithms. The exogenous strategy parameters of an evolution strategy such as μ, the population size or ν, the selective pressure, are summarized later, see Table 5.3 (the multiplier τ_0^m from Figure 2.5 is listed as c_τ in this table). Regarding the half-normal plot in Figure 2.5, τ_0^m, the multiplier for the individual and global mutation parameters, the population size μ, and the selection pressure have a significant influence on the algorithm's performance. In addition, the interactions between μ and τ_0^m, and μ and ν play a significant role.

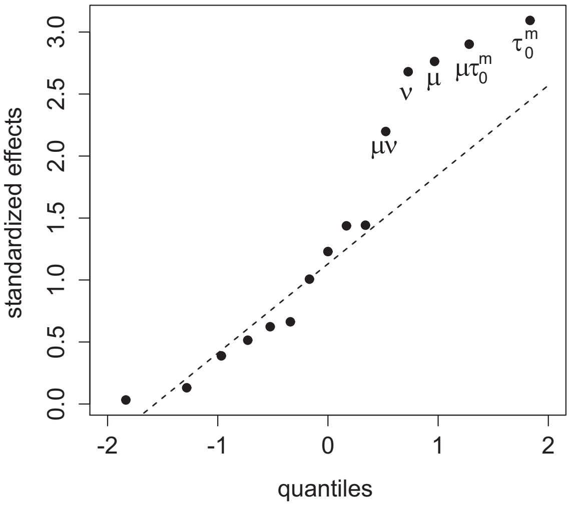


Figure 2.5: *Half-normal plot. Optimizing the exogenous strategy parameters of an evolution strategy:* τ_0^m*, the multiplier for the individual and global mutation parameters, the population size* μ*, and the selection pressure have a statistically significant influence on the algorithm's performance. In addition, the interactions between* μ *and* τ_0^m*, and* μ *and* ν *play a significant role. Source: Mehnen et al. (2004a).*

2.7.2 Scatter Plots

A *scatter plot* is a simple way to visualize data (Chambers et al., 1983; Croarkin and Tobias, 2004). It displays important information on how the data are distributed or the variables are related: Are the design variables x_1 and x_2 related? Are variables x_1 and x_2 (non) linearly related? Does the variation in x_1 change depending on x_2? Are there outliers? This information should be taken into account before any statistical model is build.
Scatter plots have been used to detect factor settings that produced outliers and to determine suitable variable ranges. Each panel in Figure 2.6 depicts a scatter plot of the response against one factor. The relationship between the function values and different levels of social parameter c_2 of a particle swarm optimization is shown in the panel down to the right. Settings with $c_2 < 2.5$ produced many outliers. Reducing the region of interest for this variable from $[0, 4]$ to $[2.5, 4]$ resulted in less outliers as can be seen in Figure 2.7. Similar results could have been obtained with box plots, design plots or other tools from exploratory data analysis. No high level statistical model assumptions are necessary to perform this analysis.

2.7.3 Interaction Plots

If the number of factor levels is low, e.g. in two factor experiments, *interaction plots* as shown in Figure 2.8, can be used to check for interactions between the factors.

Interaction plots show how pairs of factors, i.e. selection method and population size in evolution strategies, interact in influencing the response (Y). On the horizontal axis of the plot levels of the first factor are shown: Comma selection (Comma), plus selection (Plus), and threshold selection (TS). Lines are drawn for the mean of the response for the corresponding level of the interaction between selection method and selective strength (left panel) and selection method and population size (right panel). As the lines run in parallel in both panels, no interactions, which might difficult the analysis, can be detected. These figures indicate that the selection method TS improves the performance independently from the population size or selective pressure. This example was taken from Beielstein and Markon (2001).

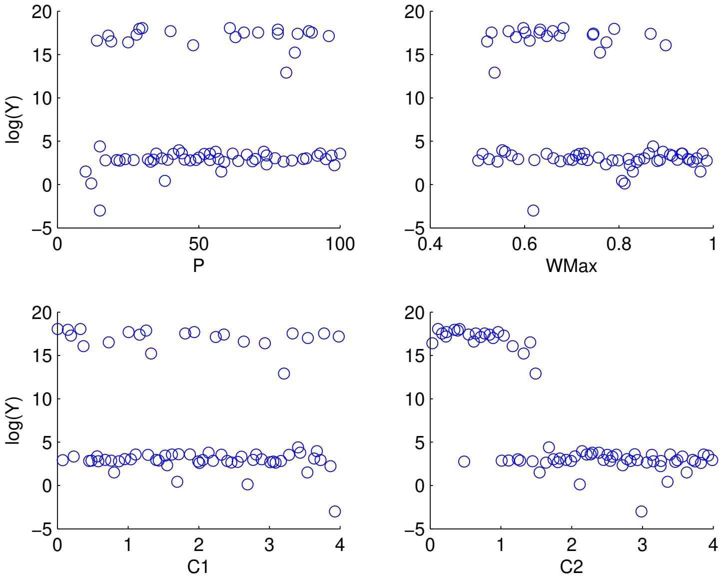


Figure 2.6: *Scatter plots of the response* $\log(y)$ *against* p, c_1, c_2, *and* $w_{\max}$. *Latin hypercube design. PSO. Rosenbrock function. The first 4 plots show the initial variable settings. Modifying the variable range of the social parameter* c_2 *from* $[0, 4]$ *to* $[2.5, 4]$ *leads to improved results (less outliers). Changes due to this modification are shown in Figure 2.7.*

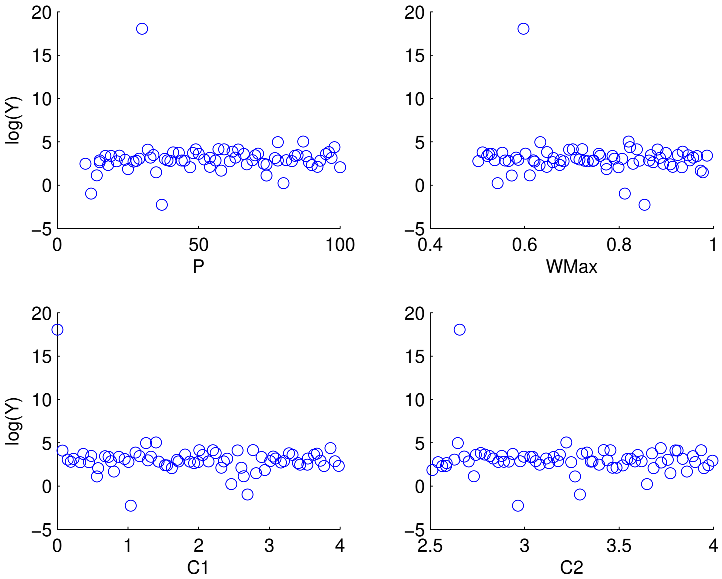


Figure 2.7: *Scatter plots of the response* $\log(y)$ *against* p, c_1, c_2, *and* $w_{\max}$. *The variable range of the social parameter* c_2 *was shrunk from* $[0, 4]$ *to* $[2.5, 4]$. *The number of outliers was reduced.*

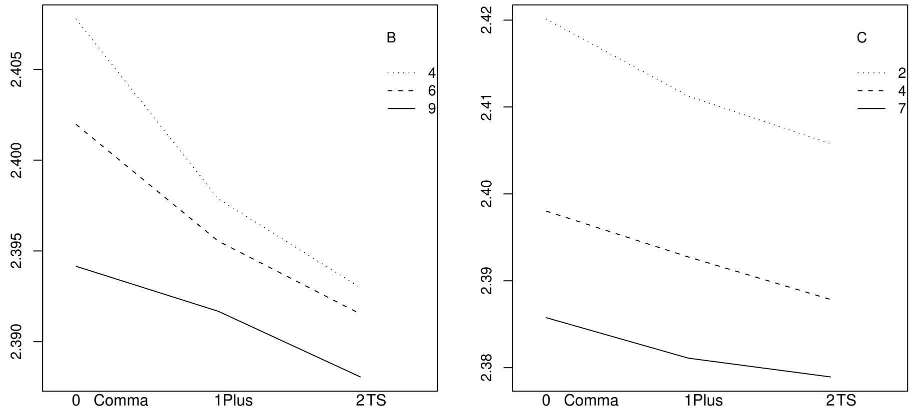


Figure 2.8: *Interaction plots. Plot of the means of the responses. The labels on the x-axis represent different selection mechanism for an evolution strategy: Comma selection (0), plus selection (1), and threshold selection (2). Selective strength (B) with levels 4, 6, and 9 and population size (C) with levels 2,4, 7 have been chosen for the comparison. Source: Beielstein and Markon (2001).*

2.7.4 Box Plots

Box plots as shown in Figure 2.9 display the distribution of a sample. They are excellent tools for detecting changes between different groups of data (Chambers et al., 1983). Let the interquartile range (IQR) be defined as the difference between the first and the third sample quartiles. Besides the three sample quartiles (the lower quartile, the median, and upper quartile), the minimum, and the maximum value, two limits are used to generate the box plots: $y_l = q_{.25} - 1.5\text{IQR}$ and $y_u = q_{.25} + 1.5\text{IQR}$. Possible outliers may lie outside the interval $[y_l, y_u]$.

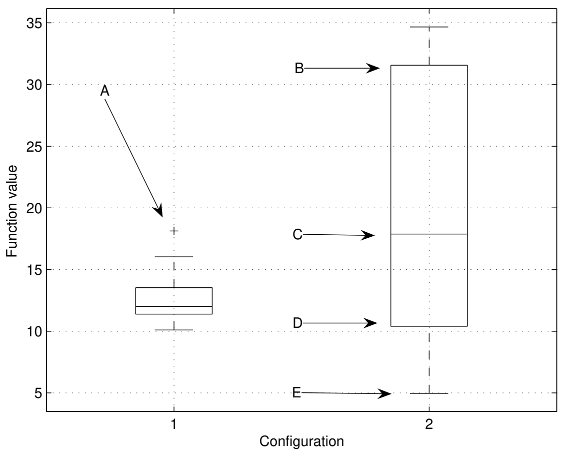


Figure 2.9: *Five elements of a box plot. This figure shows: Possible outlier A, quartiles B, C, D, and adjacent value E.*

2.8 Tree-Based Methods

Van Breedam (1995) applied *tree-based classification methods* to analyze algorithms. He used an *automatic interaction detection* (AID) technique developed by Morgan and Sonquist (1963) to determine the significant parameter settings of genetic algorithms. Breiman et al. (1984) introduced *classification and regression trees* (CART) as a "flexible non-parametric tool to the data analyst's arsenal." Tree-based methods can be deployed for screening variables and for checking the adequacy of regression models (Therneau and Atkinson, 1997). AID and CART use different pruning and estimation techniques.

The construction of regression trees can be seen as a type of variable selection (Chambers and Hastie, 1992; Hastie et al., 2001). Consider a set of design variables $X = \{x_1, \dots, x_k\}$ and a quantitative response variable Y. Design variables are called *predictor variables* in the context of CART. 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., Figure 2.11). The binary tree is build up by recursively splitting the data in each node. The tree can be read as follows: If the rule that is specified at the node is true, then take the branch to the left, otherwise take the branch to the right. The partitioning algorithm stops when the node is homogeneous or the node contains too few observations. If qualitative and quantitative design variables are in the model, then *tree-based models* are easier to interpret than linear models. The endpoint of a tree is a partition of the space of possible observations.

Tree construction (TC) comprises three phases (Martinez and Martinez, 2002):

(TC-1) In the first phase of the construction of a regression tree a large tree $T_{\max}$ is grown. The partitioning procedure requires the specification of four elements: A splitting criterion, a summary statistic to describe a node, the error of a node, and the prediction error (Therneau and Atkinson, 1997). The splitting process can be stopped when a minimum node size is reached. Consider a *node* v. A *leaf* l is any node which has no child nodes, and T_L denotes the *set of all leaves* of a tree T. A *subtree* is the tree which is a child of a node.

The summary statistic is given by the *mean of the node* $\overline{y}(v)$, that is defined as the average response of the cases that fulfill the condition in the node:

$$\overline{y}(v) = \frac{1}{n_v} \sum_{x_i \in v} y_i,$$

where n_v denotes the number of cases in this node. The *squared error of a node* is related to the variance of $y(v)$, it is defined as

$$R(v) = \frac{1}{n} \sum_{x_i \in v} \left(y_i - \overline{y}(v)\right)^2,$$

where n denotes the size of the entire sample. The *mean squared error for the tree* T is obtained by adding up all of the squared errors in all of the leaves:

$$R(T) = \sum_{l \in L} R(l).$$

The mean squared error for the tree is also referred to as the *total within-node-sum-of-squares*. As a splitting criterion, the change in the mean squared error for a split s_v is used:

$$\Delta R(s_v) = R(v) - (R(v_L) + R(v_R)),$$

where v_L and v_R denote the left and right subtrees with root node v, respectively. The best split s_v^* is the split that maximizes the change in the mean squared error $\Delta R(s_v)$.

(TC-2) The large tree $T_{\max}$ is pruned back in a second phase of the tree construction. The pruning procedure uses a cost-complexity measure:

$$R_{c_p}(T) = R(T) + c_p n_L, \quad c_p \geq 0,$$

where n_L is the number of leaves. As a large tree with leaves that contain only cases from one class has a mean squared error $R(T) = 0$, the c_p value represents the *complexity cost* per leaf. Take the tree that minimizes $R_{c_p}(T)$ to be optimal for the given value of c_p. Note that the value of $R(T)$ decreases as the size of the tree is increased, while $c_p n_L$ increases. By increasing c_p, we can move from one optimum to the next. Each move might require a reduced tree, because the optimal tree size decreases as c_p increases. The pruning procedure constructs a finite sequence of optimal subtrees such that

$$T_{\max} > T_1 > T_2 > \ldots > T_k > \ldots > T_K = \{v_1\},$$

where $\{v_1\}$ is the root node of the tree, and

$$0 = c_{p_1} < \ldots < c_{p_k} < c_{p_{k+1}} < \ldots < c_{p_K}.$$

(TC-3) Finally, the "best" tree is chosen in the selection phase from the sequence of subtrees generated in Step (TC-2). To select the right tree the *cross-validation estimate for the prediction error* $R_{\mathrm{CV}}(T_k)$ for each tree in the sequence of pruned trees is determined. An estimate of the *standard error of the cross-validation estimate of the prediction error* $s_R(T_k)$ is determined next. Let T' denote the subtree that has the smallest estimated prediction error. Its standard error is denoted as $s_R(T')$. The *one-standard error rule* (1-SE rule) selects the smallest tree T^* with

$$R_{\mathrm{CV}}(T^*) \leq R_{\mathrm{CV}}(T') + s_R(T'). \tag{2.14}$$

The tree selection procedure is depicted in Figure 2.10. The *resubstitution error*, that is calculated from the whole data set, is also shown. It gives an optimistic assessment of the relative error and should only be used to control the tree selection procedure. Not the tree with the smallest error is chosen, but the smallest tree that reaches the error-corridor of the smallest error plus one standard error (1-SE rule). The 1-SE rule chooses the tree with six nodes, since its error lies in the 1-SE corridor of the tree with seven nodes.

2.9 Design and Analysis of Computer Experiments

Matheron, the founder of Geostatistics, discovered the pioneering work of the South African school on the gold deposits of the Witwatersrand (Krige, Sichel, and de Wijs). He built the major concepts of the theory for estimating resources, i.e. Kriging (Isaaks and Srivastava, 1989). *Kriging* is an interpolation method to predict unknown values of a stochastic process and can be applied to interpolate observations from computationally expensive simulations. Our presentation follows concepts introduced in Sacks et al. (1989), Jones et al. (1998), and Lophaven et al. (2002b).

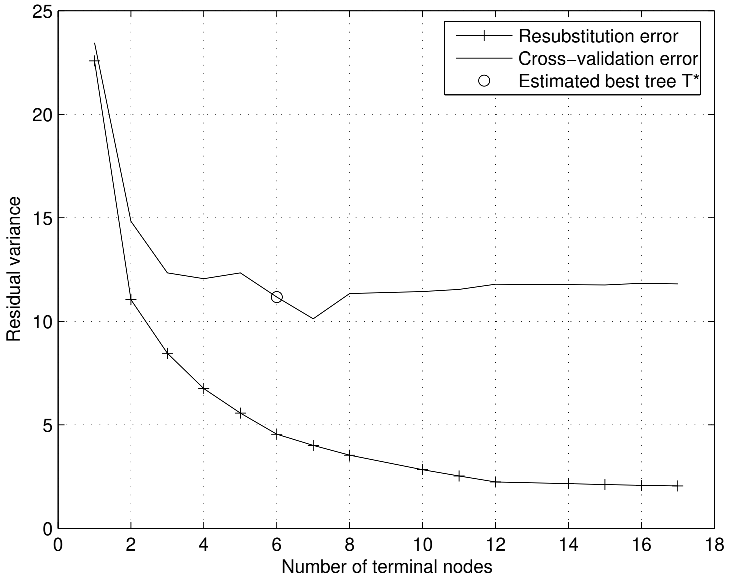


Figure 2.10: *Visualization of the 1-SE (one-standard error) selection rule to determine the right tree. The tree with the smallest cross-validation error* R_{CV} *is not chosen. The 1-SE rule chooses the tree with six nodes, because its error lies in the 1-SE corridor of the tree with the smallest error (the tree with seven nodes). The pruned tree with six nodes is shown in Figure 2.11. The full tree has 17 nodes. In addition, the resubstitution error is plotted against the number of nodes in the tree. It is monotonically decreasing, because it does not measure the costs for including additional nodes in the tree.*

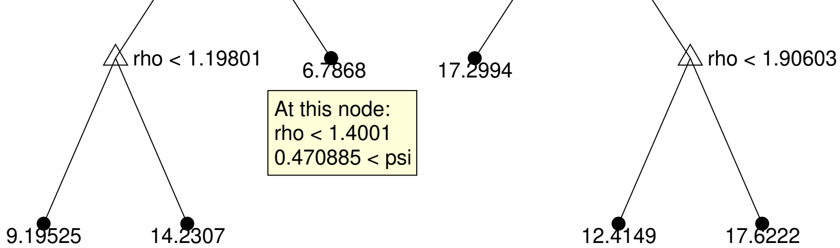


Figure 2.11: *Nelder-Mead. Pruned regression tree. The left subtree of a node contains the configurations that fulfill the condition in the node. It is easy to see that smaller* ρ *and larger* ψ *values improve the algorithm's performance.*

2.9.1 The Stochastic Process Model

The regression model

$$y = X\beta + \epsilon,$$

cf. Equation 2.11, requires that the response y and the error ϵ have the same variance. The assumption of a constant variance is unrealistic in simulation and optimization. A variance that varies with x appears to be more realistic. For example, Kleijnen (1987) reports that the standard errors of the y_i's in simulation models differ greatly. Similar ideas are presented in Jones et al. (1998), where stochastic process models as alternatives to regression models are introduced.

Consider a set of m design points $x = (x_1, \ldots, x_m)^T$ with $x_i \in \mathbb{R}^k$ as in Section 2.6. In the *design and analysis of computer experiments* (DACE) *stochastic process model*, a deterministic function is evaluated at the m design points x. The vector of the m responses is denoted as $y = (y_1, \ldots, y_m)^T$ with $y_i \in \mathbb{R}$. The process model proposed in Sacks et al. (1989) expresses the deterministic response $y(x_i)$ for a k-dimensional input x_i as a realization of a regression model $\mathcal{F}$ and a stochastic process Z,

$$Y(x) = \mathcal{F}(\beta, x) + Z(x). \tag{2.15}$$

2.9.2 Regression Models

We use q functions $f_j : \mathbb{R}^k \to \mathbb{R}$ to define the regression model

$$\mathcal{F}(\beta, x) = \sum_{j=1}^{q} \beta_j f_j(x) = f(x)^T \beta.$$

Regression models with polynomials of orders 0, 1, and 2 have been used in our experiments. The constant regression model with $q = 1$ reads $f_1(x) = 1$, the linear model with $q = k + 1$ is $f_1(x) = 1, f_2(x) = x_1, \ldots, f_{k+1}(x) = x_k$, and the quadratic model: $f_1(x) = 1$; $f_2(x) = x_1, \ldots, f_{k+1}(x) = x_k$; $f_{k+2}(x) = x_1 x_1, \ldots, f_{2k+1}(x) = x_1 x_k$; $\ldots$; $f_q(x) = x_k x_k$.

2.9.3 Correlation Models

The random process $Z(\cdot)$ is assumed to have mean zero and covariance $V(w, x) = \sigma^2 \mathcal{R}(\theta, w, x)$ with process variance σ^2 and correlation model $\mathcal{R}(\theta, w, x)$. Correlations of the form

$$\mathcal{R}(\theta, w, x) = \prod_{j=1}^{k} \mathcal{R}_j(\theta, w_j - x_j)$$

will be used in our experiments. The correlation function should be chosen with respect to the underlying process (Isaaks and Srivastava, 1989). Lophaven et al. (2002a) discuss 7 different models. Well-known examples are

$$\begin{aligned} \text{EXP} &: \quad \mathcal{R}_j(\theta, h_j) = \exp(-\theta_j |h_j|), \\ \text{EXPG} &: \quad \mathcal{R}_j(\theta, h_j) = \exp(-\theta_j |h_j|^{\theta_{k+1}}), \quad 0 < \theta_{k+1} \leq 2, \\ \text{GAUSS} &: \quad \mathcal{R}_j(\theta, h_j) = \exp(-\theta_j h_j^2), \end{aligned} \tag{2.16}$$

with $h_j = w_j - x_j$, and for $\theta_j > 0$. The exponential correlation function EXP has a linear behavior near the origin, the gaussian correlation function GAUSS has a parabolic behavior near the origin, whereas the general exponential correlation function EXPG can have both shapes. Large θ_j's indicate that function values at points in the vicinity of a point are correlated with Y at that point, whereas small θ_j's indicate that also distant data points influence the prediction at that point.

Maximum likelihood estimation (MLE) methods to estimate the parameters θ_j of the correlation functions from Equation 2.16 are discussed in Lophaven et al. (2002a). DACE methods provide an estimation of the prediction error on an untried point x, the *mean squared error* (MSE) of the predictor

$$\mathrm{MSE}(x) = E\left(\hat{y}(x) - y(x)\right). \tag{2.17}$$

2.9.4 Sensitivity Analysis

Santner et al. (2003) recommend to use a small design to determine important factor levels. After running the optimization algorithm, scatterplots of each input versus the output can be analyzed. Welch et al. (1992) advocate the use of sensitivity analysis. A screening algorithm that is similar in spirit to forward selection in classical regression analysis is used to identify important factors. Sacks et al. (1989) propose an ANOVA-type decomposition of the response into an average, main effects for each factor, two-factor interactions and higher-order interactions. A similar approach was proposed by Schonlau (1997) to plot the estimated effects of a subset x_{effect} of the x variables. We prefer the three dimensional visualizations that can be produced with the DACE toolbox (Lophaven et al., 2002b). They can be used to illustrate the interaction between two design variables and the associated mean squared error of the predictor.

2.10 Comparison

Comparing classical linear regression models and tree-based regression models, we can conclude that regression trees present results in an intuitively understandable form. The results are immediately applicable, interactions are automatically included. Regression trees can handle qualitative and quantitative factors. On the other hand, it is not guaranteed that the overall tree is optimal. The splitting criteria are only locally optimal, it is only guaranteed that each single split will be optimal. Trees can become quite large and can then make a poor intuitive sense.

If only a small set of data is available, parametric models might be advantageous that are based on strong assumptions regarding the underlying distribution or the form of the linear model.

To compare different regression techniques the following criteria might be useful: Is the technique flexible, that is can it cope with different types of variables (quantitative, qualitative) and does not require assumptions on the underlying distribution? Are the results plausible, can even complex interactions be detected? Is the method able to model gathered data and to predict new values? The availability of related literature and software packages should be judged as well.

A combination of different techniques is useful. Tree-based techniques may produce different insights than regression models. Regression trees can be used at the first stage to screen out the important factors. If only a few quantitative factors remain in the model, DACE techniques can be applied to get an exact approximation of the functional relationship between parameter settings and algorithm's performance. Sequential designs have been applied successfully during our analysis (Bartz-Beielstein and Markon, 2004; Bartz-Beielstein et al., 2004b).

2.11 Summary

The basic ideas from statistics presented in this chapter can be summarized as follows:

1. A statistical hypothesis H is a statement regarding the parameters of a distribution.
2. The procedure of taking a random sample, computing a test statistic, and rejecting (or failing to reject) a null hypothesis H is called hypothesis testing.
3. Paired data can simplify the statistical analysis.
4. Computer intensive methods enable the estimation of unknown parameters.
   (a) Monte Carlo simulations can be used to estimate unknown parameters of a known distribution.
   (b) The bootstrap requires a representative sample from the population only.
   (c) It can be used to generate plots of the observed significance if the sample distribution is unknown.
5. Standard definitions from DOE:
   (a) Factors are input parameters to be varied during the experiment. Their different values are called levels. Factors can be qualitative or quantitative.
   (b) Upper and lower bounds of each factor specify the region of interest.
   (c) Output values are called responses.
   (d) The main effect of a factor is the change in the response produced by the change in the level of this factor averaged over the levels of the other factors.
   (e) An experimental design comprises a problem design and an algorithm design.
6. The total variability in the data can be partitioned into the variability between different treatments and the variability within treatments.

$$SS_T = SS_{\text{TREAT}} + SS_{\text{E}}. \tag{2.18}$$

   This fundamental ANOVA principle is used in DOE, three-based regression, and DACE.
7. Linear regression models are based on the equation $y = X\beta + \epsilon$, where X denotes a regression matrix that represents the experimental design, ϵ is a vector of error terms, y is a vector with responses, and β is a vector that represents the model parameters.
8. Half-normal plots, scatter plots, interaction plots, and box plots are tools from exploratory data analysis to visualize the distribution of the data and possible relations between factors.

9. The three basic steps to construct a regression tree are 1. growing, 2. pruning, and 3. selecting. Tree-based methods cannot replace, but they should complement classical methods.

10. The assumption of homogeneous variance in the regression model $y = X\beta + \epsilon$ (Equation 2.11) appears to be unrealistic. DACE models include this inhomogeneity in their model specification.

11. The DACE stochastic process model expresses the deterministic response as a realization of a regression model and a random process.

12. An ANOVA-like decomposition can be used in DACE to visualize the factor and interaction effects.

Chapter 3

Problems

> Don't get involved in partial problems, but always take flight to where there is a free view over the whole single great problem, even if this view is still not a clear one.
>
> Ludwig Wittgenstein

3.1 Introduction

A well established experimental procedure in evolutionary computation and related scientific disciplines like operations research or numerical analysis to judge the performance of algorithms can be described as shown in Figure 3.1. This framework relies on the assumption that the experimenter can find the best algorithm out of a set of potential candidate algorithms. *Test suites* are commonly used to compare the performance of different optimization algorithms. We assert that results from test suites provide useful means to validate whether the implementation of an algorithm is correct (validation). They provide a starting point for further investigations.

Problems related to test suites will be considered in this chapter. We will discuss aspects related to

1. Test functions.

2. Real-world optimization problems.

3. Randomly generated test problems.

Solutions for unconstrained, global optimization problems are defined as follows. Consider a real-valued function $f : \mathbb{R}^d \to \mathbb{R}$. A *local minimizer* is a *point* x^* such that there exists an ϵ environment $U_\epsilon(x^*)$ of x^* ($\epsilon > 0$) with

$$f(x^*) \leq f(x) \quad \forall\, x \in U_\epsilon(x^*). \tag{3.1}$$

Heuristic Comparing Algorithms

1. Define a set of test functions (and an associated testing environment to specify starting points, termination criteria, etc.).
2. Perform tests.
3. Report the performances of the algorithms, for example the number of successfully completed runs. Obviously, the algorithm with the highest (expected) performance is considered best.

Figure 3.1: *Heuristic for the comparison of two algorithms.*

The related *minimization problem* is written as $\min_x f(x)$. We will consider unconstrained minimization problems only. If $f(x^*) \leq f(x)$ holds for all $x \in \mathbb{R}^d$, x^* is called a *global minimizer.* The symbol $f(x)$ denotes the (objective) *function value* at x.

Whitley et al. (1996) provide a good starting point for the discussion of test functions in evolutionary computation. The following section presents problems related to test functions discussed in the EC community.

3.2 Problems Related to Test Suites

Even in the well-structured world of mathematical optimization, a reasonable choice of test functions to evaluate the effectiveness and the efficiency of optimization algorithms is not trivial. However, even if the selection of a test problem is assumed to be unproblematic, the choice of specific problem instances can cause additional problems (a problem instance is a realization of a generic optimization problem). Hence, for a given problem many different instances can be considered. For example, varying the dimension defines different instances of the sphere function. Here we can mention two important difficulties:

- The distribution of instances might influence the algorithm's performance significantly. Although the result from Goldberg (1979) and Goldberg et al. (1982) suggested that the propositional satisfiability problem (SAT) can be solved on average in $O(n^2)$, Franco and Paull (1983) showed that this result was based on a distribution of instances "with a preponderance of easy instances" (Mitchell et al., 1992). Neither was the algorithm clever, nor was the problem easy: Goldberg sampled from the space of all problem instances without producing any hard cases.

- Increasing the dimensionality of a problem can make a test function easier. This was demonstrated for Griewangk's function, because the number of local optima decreases in number and complexity as the dimensionality increases (Whitley et al., 1996).

Whitley et al. (1996) discuss further aspects of test functions, i.e. symmetry and separability. A test function is *separable* if the global optimum can be located by optimizing each variable independently. A two dimensional function is *symmetric* if $f(x, y) = f(y, x)$

$\forall x, y \in \mathbb{R}^d$. They state that "Surprisingly, almost all of the functions in current evolutionary search test suites are separable." Therefore, they propose new non-symmetric test functions. One of these functions, the `whit` function, is listed in Table 3.1.

A generic test suite might lead to algorithms that perform well on this particular test suite only. The recommendation of many authors to define heterogeneous test suites is merely an apparent solution. To avoid useless and misleading results, it is important to understand why an algorithm performs well or not so well.

It is common practice to finish an article with presenting tabularized result data. The raw data from these tables require a correct interpretation and should not be seen as final results but as starting points for interpretation.

It can be observed that the performance of optimization algorithms crucially depends on the *starting point* $x^{(0)}$ and other start conditions. To put more emphasis on testing the robustness (effectivity), Hillstrom (1977) proposed using *random starting points*. However, random starting points may cause new difficulties, see the discussion in Section 4.6.

The *no free lunch theorem* (NFL) for search states that there does not exist any algorithm that is better than an other over all possible instances of optimization problems. However, 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 be referred to the discussion in Whitley et al. (1995), Droste et al. (2000), and Whitley et al. (2002).

The problems presented in this subsection can be solved, for example by building better test functions. But there are other, more severe objections against the concept of strategy comparison stated in Figure 3.1 as will be seen in Chapter 6.

3.3 Test Functions

Some test functions have become very popular in the EC community. Table 3.1 lists some common functions for global, unconstrained optimization. To differentiate between test functions for efficiency and effectivity (robustness), Schwefel (1975) proposed three test scenarios: The first tests were performed to analyze the rates of convergence for quadratic objective functions, the second series to test the reliability of convergence for the general non-linear case. In a third test, the dependency of the computational effort on the problem dimension for non-quadratic problems has been studied. Therefore, problem dimensions from 3 to 1000 have been used. The problem dimensions of the second scenario were relatively small.

3.3.1 Test Functions for Schwefel's Scenario 1 and 2

The following function has been used in test series one and two, see also Table 3.1:

(Sphere) Minimum $x_i^* = 0$ for $i = 1, \ldots, d$. Optimum $f^* = 0$.

3.3.2 Test Functions for Schwefel's Scenario 2

The Rosenbrock function and the Schwefel function have been used in the second test scenario.

(Rosenbrock) Minimum $x_i^* = (1, 1)$. Optimum $f^* = 0$. Starting point $x^{(0)} = (-1.2, 1)$. This is the famous two-dimensional "banana valley" function (Rosenbrock, 1960). Some

Table 3.1: *Common test functions. Based on Whitley et al. (1996). The reader is referred to Schwefel (1995) for a more detailed discussion of test functions. Test problem instances from the S-ring optimization problem are presented in Table 3.3.*

Symbol	Name	Function
`sphere`:	sphere	$\sum_{i=1}^d x_i^2$
`rosen`:	Rosenbrock	$100(x_1^2 - x_2)^2 + (1 - x_1)^2$
`step`:	step	$\sum_{i=1}^d \lfloor x_i \rfloor$
`quartic`:	quartic function with noise	$\left(\sum_i^d i x_i^4\right) + \mathcal{N}(0,1)$
`shekel`:	Shekel's foxholes	$\left(0.002 + \sum_{j=1}^{25} 1/(j + \sum_{i=1}^{2} (x_i - a_{ij})^6)\right)^{-1}$
`rast`:	Rastrigin	$10d \left(\sum_{i=1}^d \left(x_i^2 - 10\cos(2\pi x_i)\right)\right)$
`schwe`:	Schwefel	$-x \sin\left(\sqrt{\lvert x \rvert}\right)$
`grie`:	Griewangk	$1 + \sum_{i=1}^d x_i^2/4000 - \prod_{i=1}^d \left(\cos(x_i/\sqrt{i})\right)$
`whit`:	Whitley	$-x \sin\left(\sqrt{\lvert x - z \rvert}\right) - z \sin\left(\sqrt{\lvert z + x/2 \rvert}\right)$,
		with $z = y + 47$
`l1`:	L1–Norm	$\sum_i^d \lvert x_i \rvert$
`abs`:	absolute value function	$\lvert x \rvert$
`id`:	identity function	x
`boha`:	Bohachevsky	$x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7$
`bilcos`:	bisecting line cosine	$x - \cos(\pi x)$

authors use a "generalized" Rosenbrock function defined as

$$\sum_{i=1}^{d-1} \left(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right) . \tag{3.2}$$

(Schwefel) Starting point $x^{(0)} = 0$. This function is Schwefel's problem 2.26, a slightly modified variant of Schwefel's problem 2.3 (Schwefel, 1995). Both problems are one-dimensional test functions. Besides infinitely many local optima, these functions have a machine dependent *apparent global optimizer* x^*_{ap}. Schwefel reported that most algorithms located the first or highest local minimum left or right of $x^{(0)}$. A $(10, 100)$ evolution strategy was able to reach the apparent global optimum x^*_{ap} almost always. Obviously, one dimension is sufficient to demonstrate this effect. However, a d-dimensional variant ($d \geq 1$): $\sum_{i=1}^{d} -x_i \sin\left(\sqrt{|x_i|}\right)$ can be found in the literature, i.e. Whitley et al. (1996).

3.3.3 Test Functions for Schwefel's Scenario 3

The L1-norm was used in the third scenario:

L1-Norm This function is Schwefel's problem 3.4 (and problem 2.20).

These scenarios will be reconsidered in Chapter 6. Note that the experimenter's skill is needed to set up test functions for optimization scenarios as presented above.

3.4 Elevator Group Control as a Real-World Optimization Problem

Computer simulations are a suitable means to optimize many actual real-world problems. Consider e.g. a sequence of traffic signals along a certain route or elevators' movements in high-rise buildings. *Optimization via simulation* subsumes all problems in which the performance of the system is determined by running a computer simulation. As the result of a simulation run is a random variable, we cannot optimize the actual value of the simulation output, or a singular performance of the system Y. One goal of optimization via simulation is to optimize the expected performance $E[Y(x_1, x_2, \ldots, x_n)]$, where the x_i's denote the controllable input variables (Schwefel, 1979; Azadivar, 1999; Banks et al., 2001). The stochastic nature of the simulation output forces the optimization practitioner to apply different methods than in the deterministic counterparts. The stochastic output in optimization via simulation complicates the selection process in direct search methods. The efficiency of the evaluation and selection method is a crucial point, since the search algorithm may not be able to make much progress if the selection procedure requires many function evaluations.

3.4.1 The Elevator Supervisory Group Controller Problem

The construction of elevators for high-rise buildings is a challenging task. Today's urban life cannot be imagined without elevators. The elevator group controller is a central part of an elevator system. It assigns elevator cars to service calls in real-time while optimizing the overall service quality, the traffic throughput, and/or the energy consumption. The *elevator supervisory group control* (ESGC) problem can be classified as a combinatorial optimization problem (Barney, 1986; So and Chan, 1999; Markon and Nishikawa, 2002). It reveals the same complex behavior as many other stochastic traffic control problems such as materials handling systems with automated guided vehicles (AGVs). Due to many difficulties in analysis, design, simulation, and control, the elevator optimization problem has been studied for a long time. First approaches were mainly based on analytical methods derived from queuing theory. Today, *computational intelligence* (CI) methods and other heuristics are accepted as state of the art (Crites and Barto, 1998; Schwefel et al., 2003).

The elevator group controller determines the floors where the cars should go to. Passengers requesting for service can give hall calls. Since the group controller is responsible for the allocation of elevators to hall calls, a control strategy to perform this task in an optimal manner is required. The 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*.

During a day, different traffic patterns can be observed. For example, in office buildings, an *up-peak traffic* is observed in the morning, when people start to work, and, symmetrically, *down-peak traffic* is observed in the evening. Most of the day there is *balanced traffic* with much lower intensity than at peak times. *Lunchtime traffic* consists of two (often overlapping) phases where people first leave the building for lunch or head for a restaurant floor, and then get back to work (Markon, 1995). The ESGC problem subsumes the following task:

> How to assign elevators to passengers in real-time while optimizing different elevator configurations with respect to overall service quality, traffic throughput, energy consumption etc.

Figure 3.2 illustrates the dynamics in an elevator system. Fujitec, one of the world's leading elevator manufacturers, developed a controller that uses a *neural network* (NN) and a set of fuzzy controllers. The weights on the output layer of the neural network can be modified and optimized. The associated optimization problem is quite complex, because it requires the identification of globally optimal NN weights. A further analysis (not shown here) reveals that the distribution of local optima in the ESGC search space is unstructured and there are many flat plateaus. A *plateau* is a region of candidate solutions with identical function values. It can be described as follows: For a given candidate solution $x_0 \in \mathbb{R}^d$ exists an ϵ-environment $B(x_0, \epsilon)$ such that $f(x_0) = f(x)\ \forall x \in B(x_0, \epsilon)$.

The objective function values are stochastically disturbed due to the nondeterminism of service calls, and dynamically changing with respect to traffic loads.

In general, ESGC-research results are incomparable, since the elevator group control *per se* is not appropriate as a benchmark problem:

- Elevator systems have a very large number of parameters that differ widely among buildings, elevator models, manufacturers etc.
- Elevator cars have complex rules of operation, and even slight differences, e.g. in door operation or in the conditions for changing the traveling direction, can affect the system performance significantly. Even small elevator systems have a very large state space, making direct solution infeasible, thus no exact solutions are available for comparison. The sophisticated ESGC rules are usually trade secrets of the manufacturers, and cannot be made commonly available for research.

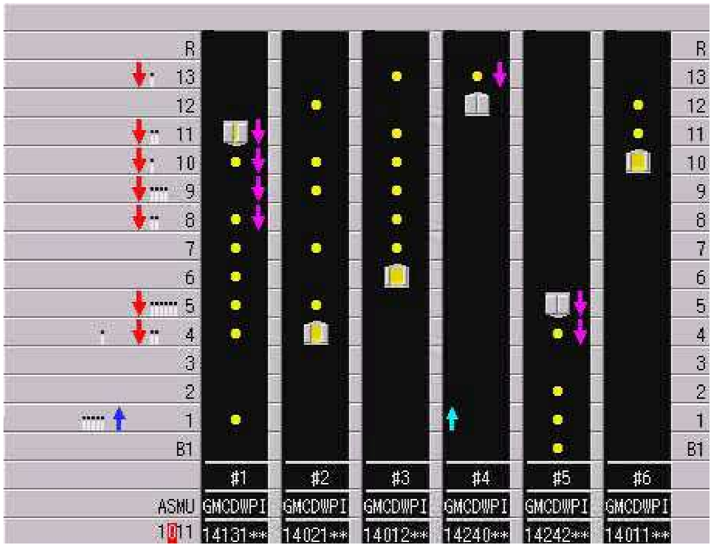


Figure 3.2: *Visualization of the dynamics in an elevator system. Fujitec's elevator simulator representing the fine model. Six elevator cars are serving 15 floors. This model is computationally expensive and has a high accuracy. Source: Beielstein et al. (2003a).*

In principle, the optimization practitioner can cope with the enormous complexity of the ESGC problem in two different ways: (1) The problem can be simplified or (2) resources can be used extensively. A parallel approach that makes extensive use of a batch-job processing system is presented in Beielstein et al. (2003b). We will concentrate on the first strategy and present a simplified ESGC model. Ideally, a simplified ESGC model should comply with the following requirements: It should enable fast and reproducible simulations and should be applicable to different building and traffic configurations. Furthermore it must be a valid simplification of a real elevator group controller and thus enable the optimization of one specific controller *policy* π and the comparison of different controller policies. The simplified model should be scalable to enable the simulation of different numbers of floors or servers. It should be extensible, so that new features (i.e. capacity constraints) can be added. Last but not least, the model is expected to favor experimental and theoretical analyses. In the next section we propose a model that conforms to all these requirements.

The approach presented here uses two models, an elevator simulator and the S-ring as a simplified model. It is related to *space mapping techniques*. Space mapping techniques iteratively update and optimize surrogate models (Bandler et al., 2004). They use two models for optimization, one fine model (Fujitec's simulator, see Figure 3.2) that is computationally expensive and has a high accuracy and one coarse (surrogate) model (the S-ring, see Figure 3.3) that is fast to solve, but less accurate. Their goal is to achieve an improved solution with a minimal number of expensive function evaluations.

3.4.2 A Simplified Elevator Group Control Model: The S-ring

When passengers give a hall call, they simply press a button. Therefore, only a one bit information for each floor is sent to the ESGC. It appears intuitively correct to map the whole state of the system to a binary string. The system dynamics is represented by a state transition table and can be controlled by a policy. The *sequential-ring model* (S-ring model) has only a few parameters: The number of elevator *cars* (also called servers) m, the number of *sites* n, and the *passenger arrival rate* p (Markon et al., 2001). A 2-bit state (s_i, c_i) is associated with each site. The s_i bit is set to 1 if a server is present on the ith floor, to 0 otherwise. Correspondingly, the c_i bit is set to 0 or 1 if there is no waiting passenger

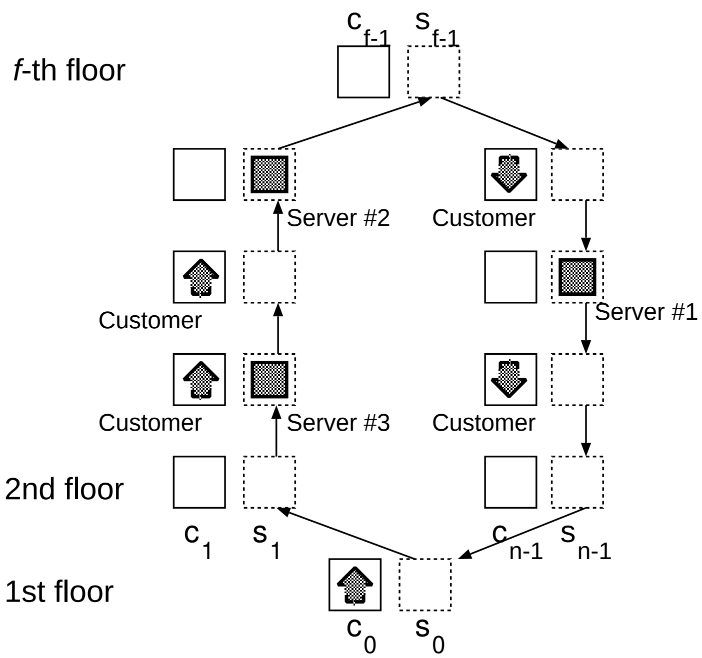


Figure 3.3: *The S-ring as an elevator system. Three cars are serving six floors (or 10 sites). The sites are numbered from 0 to $n-1$. There are $f = n/2-1$ floors. This is a coarse (surrogate) model that is fast to solve, but less accurate. Results obtained from this model should be transferable to other systems. Source: Markon et al. (2001).*

respectively at least one waiting passenger. The *state of the system* at time t is given as

$$x(t) := (s_0(t), c_0(t), \ldots, s_{n-1}(t), c_{n-1}(t)) \in \mathbb{B}^{2n}, \tag{3.3}$$

with $\mathbb{B} := \{0, 1\}$.

Example 3.1 (State of the S-ring system)
The vector $x(t) = (0, 1, 0, 0, \ldots, 0, 1, 0, 0)^T$ represents the state of the system that is shown in Figure 3.3. For example, there is a customer waiting on the first floor ($c_0 = 1$), but no server present ($s_0 = 0$). ■

A *state transition table* (Table 3.2) is used to model the dynamics in the system. The state evolution is sequential, scanning the sites from $n-1$ down to 0, and then again around from $n-1$. The up and down elevator movements can be regarded as a loop. This motivates the ring structure. At each time step, one of the sites (floor queues) is considered, where passengers may arrive with probability p.

Example 3.2 (S-ring)
Consider the situation at the third site (the up direction on the third floor) in Figure 3.3. As a customer is waiting, and a server is present, the controller has to make a decision. The car can serve the customer (take decision), or it can ignore the customer (pass decision). The former would change the values of the corresponding bits from $(1, 1)$ to $(1, 0)$, the latter from $(1, 1)$ to $(0, 1)$. ■

As the rules of operation are very simple this model is easily reproducible and suitable for benchmark testing. Despite the model's simplicity, it is hard to find the optimal policy π^* even for a small S-ring; the real π^* is not obvious, and its difference from heuristic suboptimal policies is non-trivial.

Table 3.2: *The triple $\xi(t) = (c_i, s_i, s_{i+1})$ in the first column represents the state of the current site: Customer waiting, server present, and server present on the next floor. The probability of a state change to the state $\xi(t+1)$ shown in the fourth column is given in the second column. Columns three and five denote the decision and the change in the number of sites with waiting customers, see Equation 3.4. I.e., the server has to make a decision π (to take or to pass the customer) if there is a customer waiting (1xx), and if there is a server present on the same floor (11x) but no server on the next floor (110). Columns representing configurations in which the policy affects the state of the systems are shaded dark grey.*

$\xi(t)$	Prob	$\pi(x)$	$\xi(t+1)$	$Q(t+1) - Q(t)$
000	$1-p$	$\{0,1\}$	000	0
000	p	$\{0,1\}$	100	$+1$
001	$1-p$	$\{0,1\}$	001	0
001	p	$\{0,1\}$	101	$+1$
010	$1-p$	$\{0,1\}$	001	0
010	p	0	101	-1
010	p	1	010	0
011	1	$\{0,1\}$	011	0
100	1	$\{0,1\}$	100	0
101	1	$\{0,1\}$	101	0
110	1	0	101	0
110	1	1	010	$+1$
111	1	$\{0,1\}$	011	-1

So far, the S-ring has been described as a simulation model. To use it as an optimization problem, it is equipped with an objective function. Consider the function that counts the sites with waiting customers at time t

$$Q(t) = \hat{Q}(x,t) = \sum_{i=0}^{n-1} c_i(t). \tag{3.4}$$

Then the steady-state time-average number of sites with waiting customers in the queue is

$$\mathcal{Q} = \lim_{T\to\infty} \frac{\int_0^T Q(t)dt}{T} \quad \text{with probability one.} \tag{3.5}$$

The basic optimal control problem is to find a policy π^* for a given S-ring configuration. The optimal policy minimizes the expected number of sites with waiting passengers in the system, that is the steady-state time-average as defined in Eq. 3.5. A $2n$-dimensional vector, $y \in \mathbb{R}^{2n}$, can be used to represent the policy. Let $\theta : \mathbb{R} \to \mathbb{B}$ define the Heaviside function:

$$\theta(z) = \begin{cases} 0, & \text{if } z < 0 \\ 1, & \text{if } z \geq 0, \end{cases} \tag{3.6}$$

and $x = x(t)$ be the state at time t (see Eq. 3.3). A linear discriminator, or perceptron,

$$\pi(x) = \pi(x,y) = \theta\langle y, x\rangle, \tag{3.7}$$

can be used to present the policy in a compact manner. For a given vector y that represents the policy, and a given vector x that represents the state of the system, a *take* decision occurs if $\pi(x, y) \geqslant 0$, otherwise the elevator will ignore the customer.

The most obvious heuristic policy is the greedy one: When given the choice, always serve the customer. The $2n$-dimensional vector $y_{greedy} = (1, 1, \ldots, 1)^T$ can be used to represent the greedy policy. This vector guarantees that the product in Eq. 3.7 equals 1, which is interpreted as a take decision. Rather counter-intuitively, this policy is not optimal, except in the heavy traffic ($p > 0.5$) case. This means that a good policy must bypass some customers occasionally to prevent a phenomenon that is known as *bunching*, which occurs in elevator systems when nearly all elevator cars are positioned in close proximity to each other.

The perceptron S-ring problem can serve as a benchmark problem for many optimization algorithms, since it relies on the fitness function

$$F : \mathbb{R}^{2n} \to \mathbb{R}$$

(Markon et al., 2001; Beielstein and Markon, 2002). Figure 3.4 shows the correlation between the noisy function values and the estimated function values.

Bartz-Beielstein et al. (2005b) describe the S-ring model as a partially-observable Markov decision process in detail.

3.4.3 The S-Ring Model as a Test Generator

The S-ring model can be used to generate test problem instances. An S-ring problem instance can be characterized by the number of sites, the number of elevator cars, the arrival probability, and the simulation time, see Table 3.3. A problem design specifies one or more instances of an optimization problem and related restrictions, i.e. the number of available resources (function evaluations). In addition, a computer experiment requires the specification of an algorithm design. As designs play an important role in experimentation, they will be discussed in the following chapter.

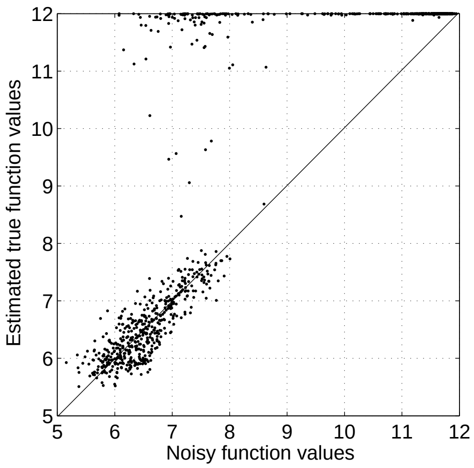


Figure 3.4: *S-ring. Estimated versus noisy function values. Test instance* `sring24` *as listed in Table 3.3. Estimated values have been gained through reevaluation, whereas noisy function values are based on one evaluation only. Points representing values from functions without noise would lie on the bisector.*

Table 3.3: *Test instances for the S-ring model*

Instance	Dimension	Number of sites	Number of elevator cars	Arrival probability	Simulation time
`sring12`	12	6	2	0.2	1000
`sring24`	24	12	4	0.2	1000
`sring36`	36	18	8	0.2	1000
`sring48`	48	24	16	0.3	1000
`sring96`	96	48	32	0.3	1000

3.5 Randomly Generated Test Problems

Although the S-ring model can be used to generate problem instances at random, these instances have been generated deterministically. Three important objections (OB) against randomly generated problem instances can be mentioned:

(OB-1) Rardin and Uzsoy (2001) illustrate subtle and insidious pitfalls that can arise from the randomness of the instance generation procedure with a simple example: To generate instances of the n-point traveling salesperson problem (TSP), $(n \times n)$ symmetric matrices of point-to-point distances are generated as follows:

> Fill the upper triangle of an n by n cost matrix with $c_{i,j}$ generated randomly (independently and uniformly) between 0 and 20. Then complete the instance by making $c_{j,i} = c_{i,j}$ in the lower triangle and setting $c_{i,i} = 0$ along the diagonal.

The mean of the cell entries $c_{i,j}$ with $i < j$ is 10 with standard deviation 5.77. If $n = 5000$ points are generated, the average tour length will be $10 \cdot 5000 = 50,000$ with standard deviation $5.77\sqrt{5000} = 408$. Nearly every feasible tour will have a length within $\pm 3 \cdot 408$ of $50,000$. Hence, "almost any random guess will yield a good solution."

(OB-2) Reeves and Yamada (1998) report that local optima of randomly generated permutation flow-shop scheduling problem instances are distributed in a *big-valley structure*, i.e. local optima are relatively close to other local optima. This big-valley structure in the search space topology is well-suited for many optimization algorithms. But do structured problem instances, that are assumed to be more realistic, possess a similar distribution? Watson et al. (1999) showed for permutation flow-shop scheduling problems that local optima are generally distributed on large plateaus of equally-fit solutions. Therefore, the assumption of big-valley structured local optima distributions does not hold for this type of problem. Whitley et al. (2002) conclude that there are differences in the performance of scheduling algorithms on random and structured instances.
The distribution of the S-ring local optima is not purely random. An analysis of the search space shows that there are plateaus of equally good solutions.

(OB-3) To separate different sources of randomness is a basic principle in statistics. Equation 2.8 describes how the total variability can be partitioned into its components:

$$SS_{\mathrm{T}} = SS_{\mathrm{TREAT}} + SS_{\mathrm{E}}.$$

If stochastic search algorithms are subject of the analysis, using randomly generated test instances will add another source of randomness to the algorithm's randomness that might complicate the analysis.

3.6 Summary

The basic ideas from this chapter can be summarized as follows:

1. Specifying test functions, performing tests, measuring performances, and selecting the algorithm with the best performance is a commonly used procedure.
2. Not only the set of test functions, but also the set of test instances has to be chosen carefully.
3. Test functions can be distinguished from real-world optimization problems.
4. Test functions should be combined with an optimization scenario.
5. The S-ring model defines a simplified elevator group control problem. It
   (a) enables fast and reproducible simulations,
   (b) is applicable to different buildings and traffic patterns,
   (c) is scalable and extensible, and
   (d) can be used as a test problem generator.
6. A problem design specifies at least one problem instance plus related restrictions.
7. Randomly generated problem instances can complicate the analysis of stochastic search algorithms.

Chapter 4

Designs

> A common mistake people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools.
>
> Douglas Adams

4.1 Introduction

This chapter discusses designs for computer experiments. Before the optimization runs are started, the experimenter has to choose the parameterizations of the optimization algorithm and one or more problem instances.

Johnson (2002) 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. An important step to make this procedure more transparent and more objective is to use design of experiments techniques. They provide an algorithmic procedure to tune the exogenous parameter settings for the algorithms under consideration before the complex real-world problem is optimized or two algorithms are compared. 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 (Box et al., 1978; Box and Draper, 1987; Kleijnen, 1987; Kleijnen and Van Groenendaal, 1992; Law and Kelton, 2000).

We will develop experimental design techniques that are well suited for parameterizable search algorithms such as evolution strategies, particle swarm optimization, or Nelder-Mead simplex algorithms. The concept of splitting experimental designs into algorithm and problem designs, which was introduced for evolution strategies in (Beielstein et al., 2001), is detailed in the following. Algorithm tuning as introduced in Chapter 6 refers to the task of finding an optimal (or improved) algorithm design for one specific problem design.

Generally speaking, two different design techniques can be distinguished: The samples can be placed either (1) on the boundaries, or (2) in the interior of the design space. The former technique is used in classical design of experiments (DOE), whereas design and analysis of computer experiments (DACE) use the latter approach. An experiment is called *sequential*, if the experimental conduct at any stage depends on the results obtained so far. Sequential approaches exist for both variants. We recommend to use factorial designs or space-filling designs instead of the commonly used one-factor-at-a-time designs. Design decisions can be based on geometric or on statistical criteria. It is still an open question, which design characteristics are important: "... extensive empirical studies would be useful for better understanding what sorts of designs perform well and for which models"(Santner et al., 2003, p. 161).

Chapter 12 in Law and Kelton (2000) provides an introduction to the use of classical DOE techniques for computer simulations, Box et al. (1978) is a classical text on experimental design. Santner et al. (2003) give a survey of designs for modern DACE methods. Giunta et al. (2003) and Simpson et al. (2004) discuss different design considerations.

4.2 Computer Experiments

Optimization runs will be treated as experiments. There are many degrees of freedom when starting an optimization run. In many cases search algorithms require the determination of parameters such as the population size in evolutionary algorithms before the optimization run is performed. From the viewpoint of an experimenter, design variables (factors) are the parameters that can be changed during an experiment. Generally, there are two different types of factors that influence the behavior of an optimization algorithm:

1. Problem specific factors, i.e. the objective function.
2. Algorithm specific factors, i.e. the population size or other exogenous parameters.

We will consider experimental designs that comprise problem specific factors and exogenous algorithm specific factors. Algorithm specific factors will be considered first. *Endogenous* can be distinguished from *exogenous parameters* (Beyer and Schwefel, 2002). The former are kept constant during the optimization run, whereas the latter, e.g. standard deviations, are modified by the algorithms during the run. Standard deviations will be referred to as *step-widths* or *mutation strengths*. Considering particle swarm optimization, step-widths and their associated directions are frequently referred to as *velocities*.

An *algorithm design* $\mathcal{D}_A$ is a set of vectors, each representing one specific setting of the design variables of an algorithm. A design can be specified by defining ranges of values for the design variables. Note that a design can contain none, one, several or even infinitely many design points.

Example 4.1 (Algorithm design)
Consider the set of exogenous strategy parameters for particle swarm optimization algorithms with the following values: Swarm size $s = 10$, cognitive parameter $c_1 \in [1.5, 2]$, social parameter $c_2 = 2$, starting value of the inertia weight $w_{\max} = 0.9$, final value of the inertia weight $w_{scale} = 0$, percentage of iterations for which $w_{\max}$ is reduced $w_{iterScale} = 1$, and maximum value of the step size $v_{\max} = 100$. This algorithm design contains infinitely many design points. ■

The *optimal algorithm design* is denoted as $\mathcal{D}_A^*$. Optimization is interpreted in a very broad sense—it can refer to the best design point x_a^* as well as the most informative design points.

Problem designs $\mathcal{D}_P$ provide information related to the optimization problem, such as the available resources (number of function evaluations) or the problem's dimension.

An *experimental design* $\mathcal{D}$ consists of a problem design $\mathcal{D}_P$ and an algorithm design $\mathcal{D}_A$. The run of a stochastic search algorithm can be treated as an experiment with a stochastic output $Y(x_a, x_p)$, with $x_a \in \mathcal{D}_A$ and $x_p \in \mathcal{D}_P$. If the random seed is specified, the output would be deterministic. This case will not be considered further, because it is not a common practice to specify the seed that is used in an optimization run. Performance can be measured in many ways, for example as the best or the average function value for n runs. One of our goals is to find a design point $x_a^* \in \mathcal{D}_A$ that improves the performance of an optimization algorithm for one problem design point $x_p \in \mathcal{D}_P$. To test the robustness of an algorithm, more than one design point can be considered.

Example 4.2 (Problem design)
Robustness can be defined as a good performance over a wide range of problem instances. A very simple example is the function `sphere`*:* $\sum_{i=1}^{d} x_i^2$ *and a set of d-dimensional starting points* $x_i^{(0)} = \left(-i, i, \ldots, (-i)^d\right)^T$, $i = 1, 2, 3$. ■

The optimization of real-world problems requires algorithms with good initial parameters, since many real-world problems are computationally expensive, e.g., *optimization via simulation* (Schwefel, 1979; Banks et al., 2001). Therefore only a few optimization runs are possible, that should be performed with good parameter settings. Optimization practitioners are interested in obtaining a good parameter setting with a minimum amount of optimization runs. The choice of an adequate parameter setting, or design, can be based on expert knowledge. But in many cases there is no such knowledge available.

4.3 Classical Algorithm Designs

In this section we will consider the following task: Determine an improved algorithm design $x_a^* \in \mathcal{D}_A$ for one fixed problem design point $x_p \in \mathcal{D}_P$.

Consider the regression model $y = X\beta + \epsilon$ that was defined in Equation 2.11 with associated regression matrix X as introduced in Equation 2.12. The regression matrix X is referred to as the design matrix in the context of experimental designs. The optimal design can be understood as the set of input vectors $X^* \subset \mathcal{D}_A$ that generates output values y that are as informative as possible with respect the exact functional relationship (Equation 2.10). Hence, the optimal algorithm design provides more information than any other algorithm design with respect to some optimality criterion. This information can be used to detect an improved design point.

The classical criteria for optimality such as D-optimality have to cope with the dependence on the model parameters. These so-called *alphabetic optimal designs* attempt to choose design points so that some measure of error in prediction, which depends on the underlying assumed model, is minimized (Federov, 1972; Box and Draper, 1987; Pukelsheim, 1993; Spall, 2003).

Example 4.3 (Optimality criteria)

1. *A design is A-optimal if it minimizes the sum of the main diagonal elements of* $(X^TX)^{-1}$*. Hence, A-optimal designs minimize the sum of the variances of the regression coefficients.*

2. *A design is said to be D-optimal if*

$$\det\left((X^TX)^{-1}\right) \tag{4.1}$$

is minimized, where X is the design matrix (Montgomery, 2001, p. 468). ■

Often, it is not trivial to formulate the experimental goals in terms of these optimal design criteria. And, "even if we can formulate the problem in this way, finding the optimal design may be quite difficult"(Santner et al., 2003, p. 124). Despite of these problems, factorial designs as one relevant and often applied type of D-optimal designs will be introduced in the following section.

Factorial Designs

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 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 (Kleijnen, 1987). Box et al. (1978) give an instructive example that explains the weakness of the classical one-factor-at-a-time design.
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 design space.

In the following, we use orthogonal designs with two levels for each factor: The corresponding factorial design with k factors requires 2^k experimental runs. Since interactions that involve many factors can be neglected in some situations, *fractional factorial designs* omit the corresponding run configurations and require only 2^{k-p} runs. Adding center points and axial points to 2^k designs leads to *central composite designs* (CCD) with axial runs (Figure 4.1). The values of factor levels can be scaled. A variable x is called *scaled* or standardized, if x ranges between -1 and $+1$.

An important objection against 2^k designs is that non-linear effects remain undiscovered. Therefore, 2^k designs are only used 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 (Montgomery, 2001).

Hence, the entry -1 in the regression matrix X (Equation 2.12) denotes a factor at its low level, and $+1$ a factor at its high level. Tab. 4.1 depicts a fractional factorial 2^{9-5}_{III} design.

In general, the following two purposes require different designs:

1. Factorial designs are used to determine which factors have a significant effect in the screening phase of the DOE.

2. To fine-tune the algorithm in the modeling and optimization phase, CCDs, which extend the factorial designs, can be used.

The number of samples in the CCD scales as 2^k, where k is the number of factors in the model. Therefore CCD should only be used in the final phase of the DOE procedure when the number of factors is very low.

Factorial designs that are commonly used in classical DOE place samples on the boundaries of the design space. The interior remains unexplored. This is due to the following model

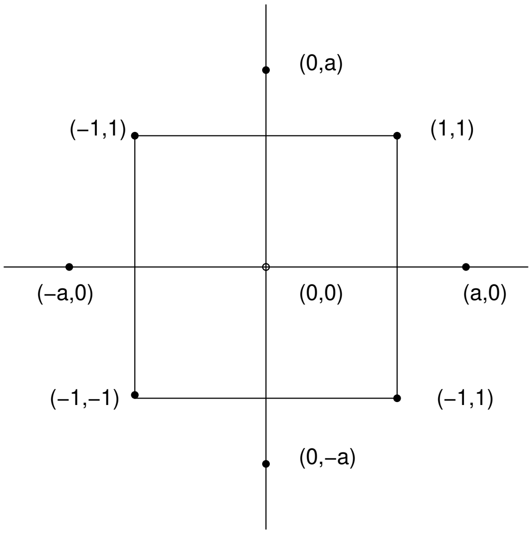


Figure 4.1: *Central composite design with axial runs for $k = 2$. The value of $a = \sqrt{k}$ gives a spherical* CCD, *that is: All factorial and axial design points are on the surface of a sphere of radius $\sqrt{k}$. Source: Montgomery (2001).*

Table 4.1: *Fractional factorial 2^{9-5}_{III} design. This design is used for screening the ES parameters. Concrete values are shown in Tab. 6.3*

	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	+	+	+	+	+	+	+	+	+

assumptions: The underlying model in the classical DOE approach can be written as

$$\tilde{y} = y + \epsilon, \tag{4.2}$$

where $\tilde{y}$ is the measured response, y the true value, and ϵ an error term. The errors are usually assumed to be independent and identically distributed. Equation 4.2 is used to model the assumption that ϵ is always present. Therefore the goal of classical DOE is to place samples in the design space so as to minimize its influence. DOE employs an approximation model

$$\hat{y} = f(x, \tilde{y}(x)), \tag{4.3}$$

where f is usually a low-order polynomial, and x denotes a sample point. We can conclude from these model assumptions that design points should be placed on the boundaries of the design space. This can be seen in Figure 4.2: The random errors remain the same in both design configurations, but the estimated linear model (dotted lines) gives a poor approximation of the true model if the samples are located in the interior of the design space (left figure). Moving the design points to the boundaries as shown in the right figure yields a better approximation of the true relationship.

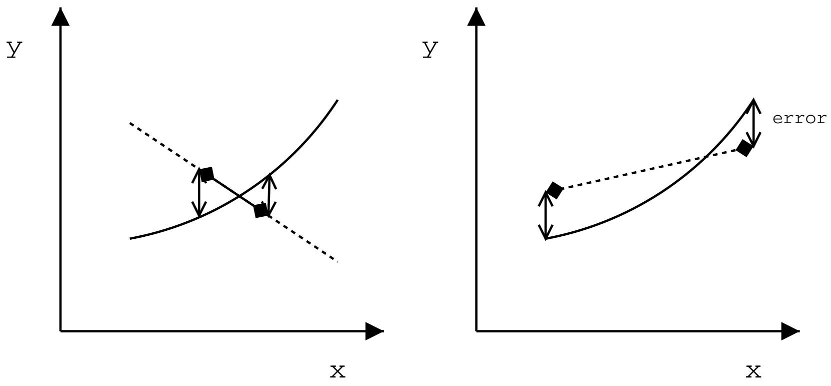
4.4 Modern Algorithm Designs

Replicate runs reduce the variance in the sample means and allow the estimation of the random error ϵ in stochastic computer experiments, cf. Equation 4.2. Modern design and analysis of computer experiments methods have been developed for deterministic computer experiments that have no random error. DACE assumes that the interesting features of the true model can be found in the whole sample space. Therefore, space filling or exploratory designs, that place a set of samples in the interior of the design space, are commonly used.

Metropolis and Ulam (1949) introduced a pseudo-Monte Carlo (MC) sampling method for computer simulations. As *Monte Carlo sampling* places samples randomly in the design space, large regions may remain unexplored. Stratified MC-sampling divides the design space into subintervals of equal probabilities and requires therefore at least 2^d samples.

Latin hypercube sampling (LHS) was developed as an alternative to Monte Carlo sampling (McKay et al., 1979). The resulting designs are called *Latin hypercube designs* (LHD). LHS is superior under certain assumptions to MC-sampling and provides a greater flexibility in choosing the number of samples. For a given design space $I = [a, b] \subseteq \mathbb{R}^d$, a Latin hypercube design with N design sites can be constructed as shown in Figure 4.3 (McKay et al.,

Figure 4.2: DOE *approximation error. The errors and true models (solid lines) are the same in both configurations. Moving the design points to the boundaries yields in a better approximation model (dotted lines). Source: (Trosset and Padula, 2000).*



Algorithm 4.1 (LHD)

1. *Partition $[a_i, b_i]$ into N subintervals of length $(b_i - a_i)/N$, $(i = 1, \ldots, d)$. The design space I is partitioned into N^d rectangles $R(j_1, \ldots, j_d)$, the index j_i denotes interval j in $[a_i, b_i]$.*

2. *Generate a random permutation $J_{i_1}, \ldots, J_{i_N}$ of $\{1, \ldots, N\}$, $(i = 1, \ldots, d)$.*

3. *Draw N samples $x_0 \sim U[R(J_{1n}, \ldots, J_{dn})]$, $(n = 1, \ldots, N)$.*

Figure 4.3: *Latin hypercube design.*

1979; Trosset and Padula, 2000). An algorithm design specifies the parameters for one specific algorithm. Latin hypercube sampling can be used to generate design points. One instance of a LHD with ten design points in two dimensions is shown in Figure 4.4. Note that LHS might result in an ill-designed arrangement of sample points, for example if the samples are placed along a diagonal as shown in Figure 4.4.

In addition, Santner et al. (2003) discuss several criteria that can be applied to generate designs by distance-based criteria, for example maxmin distance designs, or measures of the discrepancy between the empirical distribution of the set of sample points and the uniform distribution.

4.5 Sequential Algorithm Designs

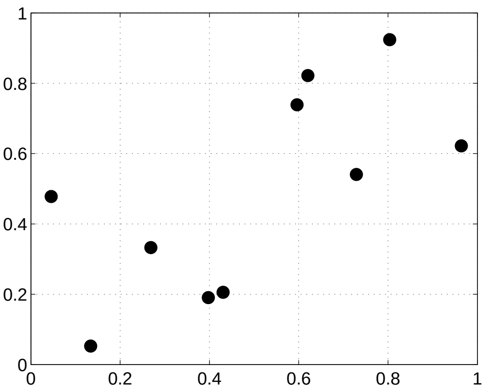
In some situations, it might be beneficial to generate the design points not at once, but *sequentially*. The selection process for further design points can include knowledge from the evaluation of previously generated design points.

Evolutionary operation (EVOP) was introduced already in the 1950s by Box (1957). The basic idea is to replace the static operation of a process by a systematic scheme of modifications (*mutations*) in the control variables. The effect of these modifications is evaluated and the process is shifted in the direction of improvement: The best *survives*. Box associated this purely deterministic process with an organic mutation-selection process. Satterthwaite (1959a,b) introduced a *random evolutionary operation* (REVOP) procedure. REVOP was rejected by Box because of its randomness.

Sequential designs can be based on several criteria, for example on the D-optimal maximization criterion as presented in Example 4.3. We will present a sequential approach that is based on a criterion developed for DACE, the *expected improvement*.

Sequential sampling approaches with adaptation have been proposed for DACE metamodels. For example, Sacks et al. (1989) classified sequential sampling approaches with and without adaptation to the existing metamodel. Jin et al. (2002) propose two sequential sampling approaches with adaptation that are not limited to DACE models.

Santner et al. (2003, p. 178) present a heuristic algorithm for unconstrained problems of global minimization. Let $y^n_{\min}$ denote the smallest known minimum value after n runs of the



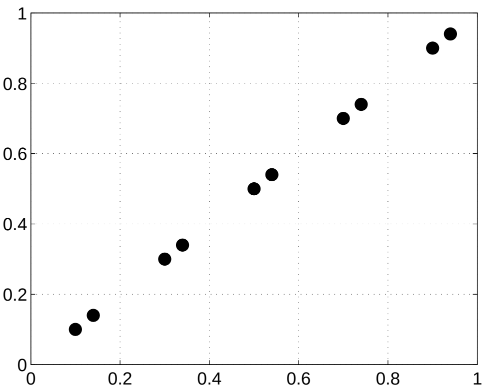


Figure 4.4: *Two* LHD *samples of ten points. Left: Typical instance of a* LHD*, right: Ill designed arrangement.*

algorithm. The *improvement* is defined as

$$\text{improvement at } x = \begin{cases} y^n_{\min} - y(x), & y^n_{\min} - y(x) > 0 \\ 0, & y^n_{\min} - y(x) \leq 0 \end{cases} \tag{4.4}$$

for $x \in \mathcal{D}_A$. As $y(x)$ is the realization of a random variable, its exact value is unknown. The goal is to optimize its expected value, the so-called *expected improvement*. The discussion in Santner et al. (2003, p. 178ff.) leads to the conclusion, that new design points are attractive "if either there is a high probability that their predicted output is below the current observed minimum and/or there is a large uncertainty in the predicted output." This result is in accordance with the experimenters' intention to avoid sites that guarantee worse results, and constituted the motivation for the EXPIMP heuristic shown in Figure 4.5 (Bartz-Beielstein and Markon, 2004; Bartz-Beielstein et al., 2004b). Next, we will discuss problem designs that consider more than just one design point of one problem design.

4.6 Problem Designs

Different instances of one optimization problem can be used to compare algorithms. The problem design and the algorithm design to be compared can be arranged in matrix form (Rardin and Uzsoy, 2001). This matrix form will be used to present performance measures that consider more than one problem design point simultaneously. These performance measures will be introduced in Section 6.3.3.

4.6.1 Initialization

It can be observed that the performance of optimization algorithms depends crucially on the *starting point* $x^{(0)}$. There are mainly two different initialization methods: Deterministic and random starts. To test the robustness of algorithms and not only their efficiency, Hillstrom proposed to use a series of random starts. Twenty random starts are considered as "a compromise between sample size and testing expense"(Hillstrom, 1977). This initialization method is nowadays often used for stochastic search heuristics such as particle swarm optimization algorithms.

Heuristic EXPIMP

1. Choose an initial design $\mathcal{D}_n$ with n points.
2. Run the algorithm at $x_i \in \mathcal{D}_n$, $i = 1, \ldots, n$, to obtain the vector of output values $y(x)$.
3. Check the termination criterion.
4. Select a new point x_{n+1} that maximizes the expected improvement, cf. Equation 4.4.
5. Run the algorithm at x_{n+1} to obtain the output $y(x_{n+1})$.
6. Set $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{x_{n+1}\}$, $n = n + 1$, and go to 3.

Figure 4.5: *Expected improvement heuristic.*

More et al. (1981) state that the use of random starts affects the reproducibility of the results. Furthermore, random starting points introduce an additional source of randomness. Since some methods of our analysis try to explain as much randomness as possible by the differences between the algorithms, this initialization method may cause unwanted side-effects that complicate the statistical analysis. Better suited for our needs are deterministic routines. We will present initialization and termination methods next.

To initialize the set of search points $X^{(0)} = \{x_1^{(0)}, \ldots, x_p^{(0)}\}$, the following methods can be used:

(DETEQ) Deterministic: Each search point uses the same vector, which is selected deterministically, i.e. $x_{\text{init}} = \mathbf{1}^T \in \mathbb{R}^d$. As this method uses only one starting point x_{init}, it is not suitable to visualize the starting points for which the algorithm converged to the optimum. This method will be referred to as initialization method DETEQ.

Example 4.4
Schwefel (1995) proposed the following initialization scheme for high dimensional non-quadratic problems:

$$x_i^{(0)} = x^* + \frac{(-1)^i}{\sqrt{d}}, \qquad \textit{for } i = 1, \ldots, d. \tag{4.5}$$

■

(DETMOD) Deterministically modified starting vectors: The algorithm can be tested with starting vectors $x^{(0)}$, $10x^{(0)}$, and $100x^{(0)}$ (More et al., 1981), or any other scheme that generates starting points deterministically. This method will be referred to as initialization method DETMOD.

(UNIRND) Uniform random starts: Every search point ($i = 1, \ldots, p$) uses the same vector $x_{\text{init}} \in \mathbb{R}^d$, where the d components are realizations of independent $U[x_l, x_u]$ random variables. This method introduces an additional source of randomness. It is suitable to visualize the starting points for which the algorithm converged to the optimum. This visualization technique is useful to get some insight into the behavior of the algorithm and will be discussed later on. This method will be referred to as initialization method UNIRND.

(NUNIRND) Non-uniform random starts: Every search point uses a different vector $x_i^{(0)}$, $(i = 1, \dots, p)$, that is $X^{(0)} = \{x_1^{(0)}, \dots, x_p^{(0)}\}$, with $x_i^{(0)} \neq x_j^{(0)}\ \forall\ i \neq j$. Each of the p vectors $x_{\text{init}} \in \mathbb{R}^d$ consists of d components that are realizations of independent $U[x_l, x_u]$ random variables. This initialization method is used by many authors. It introduces an additional source of randomness, and it is not suitable to visualize the starting points for which the algorithm converged to the optimum. This method will be referred to as initialization method NUNIRND.

Since variance reducing techniques are considered in our analysis, and we are trying to explain the variance in the results based on the fundamental ANOVA principle (Equation 2.9), we prefer a deterministic initialization scheme.

4.6.2 Termination

An algorithm run terminates, if it (or its budget) is:

(XSOL/FSOL) Solved: The problem was solved.

1. A domain convergence test becomes true when the $x_i's$ are close enough in some sense. This method will be referred to as termination criterion XSOL.
2. A function value convergence test becomes true when the function value is close enough in some sense. This method will be referred to as termination criterion FSOL.

(STAL) Stalled: The algorithm has stalled. A step size test becomes true when the step sizes are sufficiently small. This method will be referred to as termination criterion STAL.

(EXH) Exhausted: The resources are exhausted.

1. An iteration test becomes true if the maximum number of function values is exhausted.
2. A no-convergence-in-time test becomes true. This includes domain convergence and function value convergence.

This method will be referred to as termination criterion EXH.

Tests specified for the cases in which the algorithm is stalled or its budget is exhausted are called *fail tests.* Termination is as important as initialization. Even if the algorithm converges in theory, rounding errors may prevent convergence in practice. Thus, fail tests are necessary for every algorithm. Singer and Singer (2004) demonstrate the impact of the termination tests on the performance of a Nelder-Mead or simplex algorithm: “A fairly simple efficiency analysis of each iteration step reveals a potential computational bottleneck in the domain convergence test.”

4.7 Discussion: Designs for Computer Experiments

The assumption of a linear model for the analysis of computer algorithms is highly speculative. As can be seen from Figure 4.2, besides the selection of a correct regression model, the choice of design points is crucial for the whole procedure. On the other hand, DACE was introduced for deterministic computer experiments and not for the analysis of stochastic search algorithms. Performing repeated runs and taking the mean value at the design points enables the application of these techniques even for non-deterministic experiments. Determinism is "introduced through the backdoor."

Another problem that arises from DACE designs is the treatment of qualitative factors. Moreover, as Santner et al. (2003, p. 149) note:

> It has not been demonstrated that LHDs are superior to any designs other than simple random sampling (and they are only superior to simple random sampling in some cases).

Based on our experience, we can give the following recommendation: If only a few qualitative factors are relevant, then for each setting a separate LHD could be used. Otherwise, factorial design could be used to screen out those qualitative factors that have the largest effect. Latin hypercube designs can be used in the second step of the experimentation.

Despite the recommendations given in this chapter, the most frequently used strategy in practice will be the *best-guess strategy.* It works reasonably well in many situations, because it benefits from the experimenter's feeling or skill.

> In England it is still not uncommon to find in a lab a youngish technician, with no formal education past 16 or 17, who is not only extraordinarily skilful with the apparatus, but also the quickest at noting an oddity on for example the photographic plates he has prepared from the electron microscope(Hacking, 1983).

Relying upon high level experimental design theories may sometimes "help" the experimenter to miss the point.

4.8 Summary

The results from this chapter can be summarized as follows:

1. An experimental design consists of a problem design and an algorithm design.
2. Algorithm designs consider only exogenous strategy parameters.
3. Endogenous strategy parameters are modified during the run of an algorithm.
4. The objective function, its dimension, and related constraints are specified in the problem design $\mathcal{D}_P$.
5. The algorithm design $\mathcal{D}_A$ defines the set of exogenous strategy parameters of the algorithm, for example the swarm (population) size of a particle swarm optimization (PSO).
6. The task of searching for an optimized algorithm design for a given problem design is called algorithm tuning.

7. We do not recommend to use one-factor-at-a-time designs, because they fail to discover any possible interaction between the factors.

8. Factorial designs are widely used designs from classical DOE. Design points are placed on the boundaries of the design space.

9. Latin hypercube designs are popular designs for modern DACE. These designs are space filling: Design points are placed in the interior of the design space.

10. Sequential designs can be constructed for both, classical and modern designs.

11. Designs of test problems specify one specific problem instance. This specification comprises the starting conditions and the termination criteria.

12. LHDs are widely spread not because they are superior, but because they are easy to implement and the underlying design principles are comprehensible. Only seven words are necessary to explain the design principle: "Place eight non-attacking castles on a chess-board."

Chapter 5

Search

> The alchemists in their search for gold discovered many other things of greater value.
>
> Arthur Schopenhauer

5.1 Introduction

This chapter describes search algorithms for unconstrained optimization. The focus lies on the determination of their exogenous strategy parameters (design variables) to define the associated algorithm design. A short description of these algorithms is given, too.

We distinguish deterministic from stochastic search algorithms. Methods that can be found in standard books on continuous optimization such as Noceda and Wright (1999) are characterized here as deterministic optimization algorithms. Stochastic or random strategies can be defined as methods "in which the parameters are varied according to probabilistic instead of deterministic rule"(Schwefel, 1995, p. 87). If the function is continuous in its first derivative, gradient methods are usually more efficient than direct methods. Direct methods use only function evaluations. There are deterministic, for example the simplex search of Nelder and Mead, and stochastic direct search algorithms, for example evolution strategies.

5.2 Deterministic Optimization Algorithms

5.2.1 Nelder and Mead

The *Nelder-Mead simplex* (NMS) algorithm was motivated by the observation that $(d + 1)$ points are adequate to identify a downhill direction in a d-dimensional landscape (Nelder and Mead, 1965). However, $(d + 1)$ points define also a non-degenerated simplex in $\mathbb{R}^d$. Thus, it seemed a good idea to exploit a simplex for probing the search space, using only function values (Lewis et al., 2000).

Nelder and Mead incorporated a set of moves that enhance the algorithm's performance, namely *reflection*, *expansion*, *contraction*, and *shrinkage*. A new point is generated at each iteration. Its function value is compared to the function values at the vertices of the simplex.

One of the vertices is replaced by the new point. Reflection reflects a vertex of the simplex through the centroid of the opposite face. Expansion allows the algorithm to take a longer step from the reflection point (centroid) towards the reflected vertex, while contraction halves the length of the step, thereby resulting in a more conservative search. Finally, shrinkage reduces the length of all edges that are adjacent to the best vertex, i.e., the vertex with the smallest function value. Thus, there are four design variables to be specified, namely the coefficients of reflection ρ, expansion χ, contraction γ, and shrinkage σ. Default settings of these parameters are reported in Table 5.1. NMS is considered to be quite a robust but relatively slow algorithm that works reasonably well even for non-differentiable functions (Lagarias et al., 1998).

The *MATLAB* function `fminsearch` has been used to perform the experiments. It uses the following values for the design variables: $\rho = 1$; $\chi = 2$; $\gamma = 0.5$; $\sigma = 0.5$.

5.2.2 Variable Metric

The variable metric method is a quasi-Newton method. Quasi-Newton methods build up curvature information. Let H denote the hessian, c a constant vector, and b a constant, then a quadratic model problem formulation of the form

$$\min_x \frac{1}{2} x^T H x + c^T + b$$

is constructed. If the partial derivatives of x go to zero, that is

$$\nabla f(x^*) = Hx^* + c = 0$$

the optimal solution for the quadratic problem occurs. Hence

$$x^* = -H^{-1}c.$$

Quasi-Newton methods avoid the numerical computation of the inverse hessian H^{-1} by using information from function values $f(x)$ and gradients $\nabla f(x)$ to build up a picture of the surface to be optimized. The *MATLAB* function `fminunc` uses the formula of Broyden, Fletcher, Goldfarb, and Shanno (BFGS) to approximate H^{-1}. The gradient information is derived by partial derivatives using a numerical differentiation via finite differences. A line search is performed at each iteration in the direction

$$-H_k^{-1} \cdot \nabla f(x_k).$$

Figure 5.1 illustrates the solution path of the BFGS method on Rosenbrock's function. The method converges to the minimum after 140 function evaluations.

Table 5.1: *Default settings (algorithm design) of the exogenous parameters of the NMS algorithm. This design is used in the MATLAB optimization toolbox (Lagarias et al., 1998).*

Symbol	Parameter	Range	Default
ρ	reflection	$\rho > 0$	1.0
χ	expansion	$\chi > \max\{1, \rho\}$	2.0
γ	contraction	$0 < \gamma < 1$	0.5
σ	shrinkage	$0 < \sigma < 1$	0.5

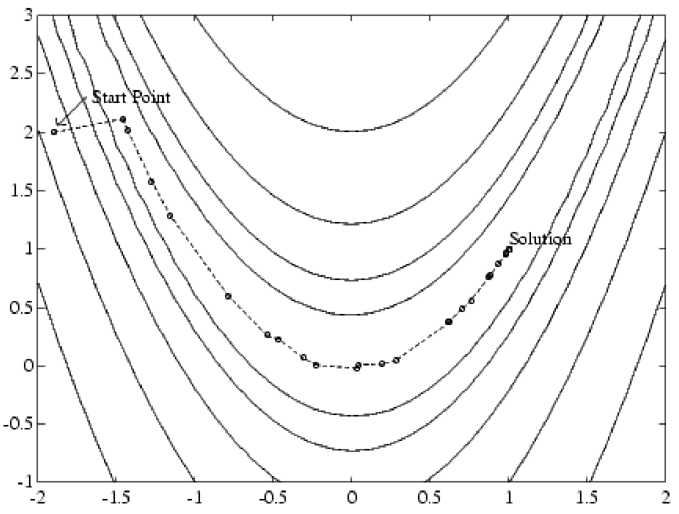


Figure 5.1: *BFGS method on Rosenbrock's function. Source: Coleman and Zhang (2004).*

5.3 Stochastic Search Algorithms

5.3.1 The Two Membered Evolution Strategy

The two membered evolution strategy, or $(1 + 1)$ ES, is included in our overview for three reasons.

1. It is easy to implement.
2. It requires only a few exogenous parameters.
3. It defines a standard for comparisons. Why use more complex algorithms if the $(1 + 1)$ ES can solve the problem effectively and efficiently?

Therefore many optimization practitioners apply the $(1+1)$ ES to their optimization problem. Schwefel (1995) describes this algorithm as "the minimal concept for an imitation of organic evolution". Let f denote an objective function $f : \mathbb{R}^d \to \mathbb{R}$ to be minimized. The rules of an $(1 + 1)$ ES can be described as shown in Figure 5.2. The standard deviation σ is interpreted as the mean *step length*. The ratio of the number of the successes to the total number of mutations, the so-called *success rate* s_r, might be modified as well as the factor by which the variance is reduced or increased, the so-called *step size adjustment factor* s_a. A related algorithm design is shown in Table 5.2.

Figure 5.3 shows the 1/5 success rule derived by Rechenberg while analyzing the $(1+1)$ ES on two basically different objective functions for selecting appropriate step lengths (Rechenberg, 1973).

A more precise formulation is required to implement the 1/5 success rule. "From time to time during the optimization run" can be interpreted as "after every s_n mutations." We analyze the following two variants to implement the 1/5 rule:

(INTV) A success counter $c \in \mathbb{N}$ is initialized at iteration $t = 1$. If a successful mutation occurs, c is increased. Every s_n iterations, the success rate is determined as c/s_n and c is set to zero. This variant will be referred to as the interval 1/5 update rule (INTV).

Procedure $(1+1)$-ES.

Initialization: Initialize the generation counter: $g = 0$. Determine a point $X_1^{(g)}$ and a standard deviation $\sigma^{(g)}$ with associated position vector $x_1^{(g)} \in \mathbb{R}^d$. Determine the function value $y_1 = f(x_1^{(g)})$.

`repeat`

Mutation: Generate a new point $X_2^{(g)}$ with associated position vector $x_2^{(g)}$ as follows:

$$x_2^{(g)} = x_1^{(g)} + z, \tag{5.1}$$

where z is a d-dimensional vector. Each component of z is the realization of a normal random variable Z with mean zero and standard deviation $\sigma^{(g)}$.

Evaluation: Determine the function value $y_2 = f(x_2^{(g)})$.

Selection: Accept $X_2^{(g)}$ as $X_1^{(g+1)}$ if its function value does not exceed that of $X_1^{(g)}$:

$$y_2 < y_1, \tag{5.2}$$

otherwise retain $X_1^{(g)}$ as $X_1^{(g+1)}$.

Adaptation: Update $\sigma^{(g)}$. Increment g.

`until` some stopping criterion is satisfied.

Figure 5.2: *The two membered evolution strategy or* $(1+1)$*-ES.*

1/5 Success Rule From time to time during the optimization obtain the frequency of successes, i.e., the ratio of the number of the successes to the total number of trials (mutations). If the ratio is greater than 1/5, increase the variance, if it is less than 1/5, decrease the variance.

Figure 5.3: *Heuristic Rule: 1/5 Success Rule.*

(CONT) A success vector $v \in \mathbb{B}^{s_n}$ is initialized at iteration $t = 1$. If a successful mutation occurs at iteration $t \mod s_n$, the t-th bit is set to 1. After an initialization phase of s_n iterations, the success rate is determined in every generation as

$$\sum_{k=1}^{s_n} v_k / s_n.$$

This variant will be referred to as the continuous 1/5 update rule (CONT).

A coding of the two membered evolution strategy and an in-depth discussion of evolution strategies and other direct search methods can be found in Schwefel's seminal book "Evolution and 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"(Schwefel, 1977), and translated into English four years later (Schwefel, 1981).

5.3.2 Multimembered Evolution Strategies

An ES-algorithm run can be described briefly as follows: The parental population is *initialized* at time (generation) $g = 0$. 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. After evaluation, a *selection* procedure is performed to determine the next parent population. The populations created in the iterations of the algorithm are called *generations* or *reproduction cycles*. A termination criterion is tested. If this criterion is not fulfilled, the generation counter (g) is incremented and the process continues with the generation of the next offspring.

We consider the parameters or control variables from Table 5.3. This table shows typical parameter settings. Bäck (1996) 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.

The reader is referred to Bartz-Beielstein (2003) for a detailed description of these parameters. Schwefel et al. (1995) and Beyer and Schwefel (2002) provide a comprehensive introduction to this special class of EA.

5.3.3 Particle Swarm Optimization

The main inspiration, which led to the development of particle swarm optimization (PSO) algorithms, was the flocking behavior of swarms and fish shoals (Kennedy and Eberhart,

Table 5.2: *Algorithm design, AD, for the two membered evolution strategy.*

Symbol	Parameter	Range	Default
s_n	adaptation interval	$\mathbb{N}$	100
s_r	1/success rate	$\mathbb{R}_+$	5
s_a	step size adjustment factor	$\mathbb{R}_+$	0.85
$\sigma^{(0)}$	starting value of the step size σ, see Figure 5.2	$\mathbb{R}_+$	1
$s_{1/5}$	step size update rule	{INTV, CONT }	CONT

Table 5.3: *Default settings of exogenous parameters of a "standard" evolution strategy. Source: Bäck (1996). Bäck does not recommend to use this "standard" without reflection. Problems may occur, when these "standards" are blindly adopted and not adjusted to the specific optimization problem.*

Symbol	Parameter	Range	Default
μ	Number of parent individuals	$\mathbb{N}$	15
$\nu = \lambda/\mu$	Offspring-parent ratio	$\mathbb{R}_+$	7
$\sigma_i^{(0)}$	Initial standard deviations	$\mathbb{R}_+$	3
n_σ	Number of standard deviations. d denotes the problem dimension	$\{1, d\}$	1
c_τ	Multiplier for individual and global mutation parameters	$\mathbb{R}_+$	1
ρ	Mixing number	$\{1, \mu\}$	2
r_x	Recombination operator for object variables	$\{i, d\}$	d (discrete)
r_σ	Recombination operator for strategy variables	$\{i, d\}$	i (intermediary)
κ	Maximum age	$\mathbb{R}_+$	1

1995). PSO has been applied to numerous simulation and optimization problems in science and engineering (Kennedy and Eberhart, 2001; Parsopoulos and Vrahatis, 2002, 2004). PSO's convergence is controlled by a set of design variables that are usually either determined empirically or set equal to widely used default values.

PSO belongs to the class of stochastic, population-based optimization algorithms (Kennedy and Eberhart, 2001). It exploits a population of individuals to probe the search space. In this context, the population is called a *swarm* and the individuals are called *particles*. Each particle moves with an adaptable velocity within the search space, and it retains in a memory the best position it has ever visited.

There are two main variants of PSO with respect to the information exchange scheme among the particles. In the *global* variant, the best position ever attained by all individuals of the swarm is communicated to all the particles at each iteration. In the *local* variant, each particle is assigned to a neighborhood consisting of prespecified particles. In this case, the best position ever attained by the particles that comprise a neighborhood is communicated among them. Neighboring particles are determined rather based on their indices than their actual distance in the search space. Clearly, the global variant can be considered as a generalization of the local variant, where the whole swarm is considered as the neighborhood for each particle. In the current work we look at the global variant only.

Assume a d-dimensional search space, $S \subset \mathbb{R}^d$, and a swarm consisting of s particles. The i-th particle is a d-dimensional vector,

$$x_i = (x_{i1}, x_{i2}, \dots, x_{id})^T \in S.$$

The velocity of this particle is also a d-dimensional vector,

$$v_i = (v_{i1}, v_{i2}, \dots, v_{id})^T.$$

The best previous position encountered by the i-th particle (i.e., its memory) in S is denoted by

$$p_i^* = (p_{i1}^*, p_{i2}^*, \dots, p_{id}^*)^T \in S.$$

Assume b to be the index of the particle that attained the best previous position among all the particles in the swarm, and t to be the iteration counter.

Particle Swarm Optimization with Inertia Weights

Then, the resulting equations for the manipulation of the swarm are (Eberhart and Shi, 1998),

$$v_i(t+1) = wv_i(t) + c_1 r_1 \left(p_i^*(t) - x_i(t)\right) + c_2 r_2 \left(p_b^*(t) - x_i(t)\right), \tag{5.3}$$
$$x_i(t+1) = x_i(t) + v_i(t+1), \tag{5.4}$$

where $i = 1, 2, \ldots, s$; w is a parameter called the *inertia weight*; c_1 and c_2 are positive constants, called the *cognitive* and *social* parameter, respectively; and r_1, r_2 are vectors with components uniformly distributed in $[0, 1]$. All vector operations are performed componentwise.

Usually, the components of x_i and v_i are bounded as follows,

$$x_{\min} \leqslant x_{ij} \leqslant x_{\max}, \quad -v_{\max} \leqslant v_{ij} \leqslant v_{\max}, \quad j = 1, \ldots, n, \tag{5.5}$$

where $x_{\min}$ and $x_{\max}$ define the bounds of the search space, and $v_{\max}$ is a parameter that was introduced in early PSO versions to avoid swarm explosion that was caused by the lack of a mechanism for controlling the velocity's magnitude. Although the inertia weight is such a mechanism, empirical results have shown that using $v_{\max}$ can further enhance the algorithm's performance. Table 5.4 summarizes the design variables of particle swarm optimization algorithms.

Experimental results indicate that it is preferable to initialize the inertia weight with a large value, in order to promote global exploration of the search space, and gradually decrease it to get more refined solutions. Thus, an initial value around 1 and a gradual decline towards 0 is considered a proper choice for w. This scaling procedure requires the specification of the maximum number of iterations $t_{\max}$. Bartz-Beielstein et al. (2004a) illustrate a typical implementation of this scaling procedure.

Proper fine-tuning of the parameters may result in faster convergence and alleviation of local minima (Bartz-Beielstein et al., 2004a; Eberhart and Shi, 1998; Beielstein et al., 2002b; Bartz-Beielstein et al., 2004b). Different PSO versions, such as PSO with constriction factor, have been proposed (Clerc and Kennedy, 2002).

Table 5.4: *Default algorithm design $x_{PSO}^{(0)}$ of the PSO algorithm. Similar designs have been used in Shi and Eberhart (1999) to optimize well-known benchmark functions.*

Symbol	Parameter	Range	Default	Constriction
s	swarm size	$\mathbb{N}$	40	40
c_1	cognitive parameter	$\mathbb{R}_+$	2	1.494
c_2	social parameter	$\mathbb{R}_+$	2	1.494
$w_{\max}$	starting value of the inertia weight w	$\mathbb{R}_+$	0.9	0.729
w_{scale}	final value of w in percentage of $w_{\max}$	$\mathbb{R}_+$	0.4	1.0
$w_{\text{iterScale}}$	percentage of iterations, for which $w_{\max}$ is reduced	$\mathbb{R}_+$	1.0	0.0
$v_{\max}$	maximum value of the step size (velocity)	$\mathbb{R}_+$	100	100

Particle Swarm Optimization with Constriction Coefficient

In the constriction factor variant, Equation (5.3) reads,

$$v_i(t+1) \quad = \quad \chi \left[v_i(t) + c_1 r_1 \left(p_i^*(t) - x_i(t) \right) + c_2 r_2 \left(p_b^*(t) - x_i(t) \right) \right], \tag{5.6}$$

where χ is the *constriction factor* (Kennedy, 2003).

Equations (5.3) and (5.6) are algebraically equivalent. In our experiments, the so-called *canonical PSO* variant proposed in Kennedy (2003), which is the constriction variant defined by Equation (5.6) with $c_1 = c_2$, has been used. The corresponding parameter setting for the constriction factor variant of PSO is reported in the last column (denoted as "Constriction") of Table 5.4, where χ is reported in terms of its equivalent inertia weight notation, for uniformity reason. Shi (2004) gives an overview of current PSO variants.

5.4 Summary

The ideas presented in this chapter can be summarized as follows:

1. An algorithm design consists of one or more parameterizations of an algorithm. It describes exogenous strategy parameters that have to be determined before the algorithm is executed.

2. The *MATLAB* function `fminunc`, which implements a quasi-Newton method, has been presented as an algorithm that can be run without specifying exogenous strategy parameters.

3. Exogenous strategy parameters have been introduced for the following stochastic and deterministic optimization algorithms:

   (a) Nelder-Mead.

   (b) Evolution strategies.

   (c) Two particle swarm optimization variants have been introduced: The inertia weight version (PSO) and the constriction factor version (PSOC).

Table 5.5: *Default settings of the exogenous parameters of PSO with constriction factor. Recommendations from Clerc and Kennedy (2002).*

Symbol	Parameter	Range	Default
s	swarm size	$\mathbb{N}$	40
χ	constriction coefficient	$\mathbb{R}_+$	0.729
φ	multiplier for random numbers	$\mathbb{R}_+$	4.1
$v_{\max}$	maximum value of the step size (velocity)	$\mathbb{R}_+$	100

Chapter 6

Comparison

> What is man in nature? A nothing in comparison with the infinite, an all in comparison with the nothing—a mean between nothing and everything.
>
> Blaise Pascal

6.1 Introduction

Algorithms have endogenous and exogenous parameters. Exogenous parameters must be specified before the algorithm is started, endogenous parameters can evolve during the optimization process, i.e. in self-adaptive evolution strategies (Beyer and Schwefel, 2002). Usually, the adaptation of endogenous parameters depends on exogenous parameters. By varying the values of the exogenous parameters the experimenter can get some insight into the behavior of an algorithm.

Exogenous parameters will be referred to as *design variables* in the context of statistical design and analysis of experiments. The parameter values chosen for the experiments constitute an algorithm design $\mathcal{D}_A$ as introduced in Section 4.2. A design point $x_a \in \mathcal{D}_A$ represents one specific parameter setting.

> Algorithm *tuning* can be understood as the *process of finding the optimal* design point $x_a^* \in \mathcal{D}_A$ for a given problem design $\mathcal{D}_P$.

The tuning procedure leads to results, that are tailored for one specific algorithm-optimization problem combination. In a similar manner as Naudts and Kallel (2000) mention "the non-sense 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. A problem being PSO easy may be ES hard, and vice versa.

Tuning enables a fair comparison of two or more algorithms that should be performed prior to their comparison. This should provide an equivalent budget—for example a number of function evaluations or an overall runtime—for each algorithm.

It is crucial to formulate the goal of the tuning experiments precisely. Tuning was introduced as an optimization process. However, in many real-world situations, it is not possible or not desired to find the optimum. Assumptions, or *boundary conditions*, that are necessary for optimization have been analyzed in operations research (OR). These assumptions comprise conditions such as (1) well defined goals, (2) stable situations and decision maker's values, or (3) an exhaustive number of alternatives. The review of these conditions demonstrates that "outside the limited-context problems presented in laboratory studies" (Klein, 2002, p.113) only very few decision problems permit optimization. The so-called *fiction of optimization* is discussed in Section 6.2. Progressive deepening, a strategy used by chess grandmasters that is described in de Groot (1978), can be used as a "vehicle for learning", whereas decision analysis is a vehicle for calculating. This classification resembles Mayo's differentiation between NPT and NPT*. The final decision whether a solution is optimal is similar to the final decision in a statistical test. Conditions required to make an optimal choice are considered in Section 6.3. This discussion is also relevant for the specification of performance measures in evolutionary computation. 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, or the number of iterations required to obtain the results.

Section 6.4 demonstrates how the classical design of experiments (DOE) approach can be used to tune algorithms. It consists of three steps: (1) Screening, (2) modeling, and (3) optimization.

In Section 6.5, a stochastic approach from design and analysis of computer experiments (DACE) is presented. This approach assumes that the correlation between errors is related to the distance between the sampling points, whereas linear regression used in DOE assumes that the errors are independent (Jones et al., 1998). Both approaches require different designs and pose different questions. In classical DOE, it is assumed that the data come from sources that are disturbed by random error. Designs with two or three levels only for each factor are used to build the corresponding regression models. DACE methods employ designs that vary each factor over many levels.

The tuning approach presented in this chapter has been successfully applied to several optimization tasks, for example in evolutionary optimization of mould temperature control strategies (Mehnen et al., 2004a), digital circuit design using evolutionary algorithms (Beielstein et al., 2002a), or elevator group control (Beielstein et al., 2003a; Bartz-Beielstein et al., 2005b). Bartz-Beielstein (2003) uses the classical DOE approach to compare a variant of simulated annealing (Belisle, 1992) to an evolution strategy. An approach that combines classical DOE techniques, regression trees and DACE was shown in Bartz-Beielstein and Markon (2004). In Section 6.5.2 we develop a related process, the so-called *sequential parameter optimization* (Bartz-Beielstein et al., 2004c).

We have also applied the tuning procedure to multi-criteria optimization problems (Bartz-Beielstein et al., 2003a; Mehnen et al., 2004a,b; Weinert et al., 2004; Bartz-Beielstein and Naujoks, 2004; Bartz-Beielstein et al., 2004d). But if not mentioned explicitly, we will consider single criteria optimization problems as defined in Equation 3.1.

Klein's viewpoint is based on the approach of bounded rationality (Simon, 1955; Rubinstein, 1998; Klein, 2002). Montgomery (2001) discusses classical DOE techniques, whereas Santner et al. (2003) give an introduction to DACE.

6.2 The Fiction of Optimization

Tuning is an optimization problem. Many researchers describe optimization as the attempt to select the option with the highest expected utility (maximization). Optimization relies on a number of very restrictive assumptions. No serious researcher would claim that these assumptions will be met in any setting, "with the possible exception of the laboratory or casino"(Klein, 2002). Uncertainty, limited time, and restricted financial resources, are only some of the reasons that prevent the determination of an optimal solution.

But why does, despite of these obvious problems, the dream of optimization linger as a gold standard for many researchers? Klein notes that the agenda for researchers is dictated by the mathematical formulation of expected utility: "... to find ways to translate decisions into the appropriate formalism." Deviations from this concept of maximization are seen as defects, that can be eliminated. "Because maximization is based on mathematical proofs, these theorems act as a bedrock."

Klein questions the value of expected utility for understanding decision making. Instead of presenting a definition of optimization, he mentions important objections against commonly used ideas related to optimizations that are also relevant for comparing algorithms.

1. Optimization does not only refer to the outcome. It is important to "provide accurate and reliable inputs to the analysis". The optimization process plays an important role.
2. It is not obvious whether the optimization refers either to the absolute best, the best solution given the data provided, or the best solution given all data that can be provided.
3. Stopping rule: "If we try to consider every relevant factor, we may not finish the analysis in a finite amount of time."
4. Suboptimal strategies are sometimes preferred in the engineering community, they are more robust than the optimal solution.

To define a measure that judges the performance of an algorithm, certain assumptions (boundary conditions) have to be fulfilled. Following Klein (2002), we will discuss some boundary conditions that have been compiled by decision researchers.

Boundary Conditions

The first assumption requires the goals to be well defined and specified in quantitative terms. This assumption appears to be unproblematic, because a performance measure can easily be defined. It is not a problem to find some performance measure—but it is a problem to find an appropriate one. Many performance measures can be defined, for example the average function value from n optimization runs, the minimum value from these runs, or the median.

Other criteria demand that the decision maker's values as well as the optimization situation must be stable. However, the decision maker might gain new insight into the problem during the optimization. The optimization goal might be redefined due to this enhanced knowledge.

An other criterion demands that the decision maker is restricted to selections between options. But a typical decision maker is not only passively executing the experiments. Learning, even by accident, may occur. New ideas for improved algorithms can come up.

One criterion requires that the number of alternatives generated must be exhaustive and that the options must be properly compared to each other. We cannot test every single algorithm configuration and every possible problem instance. Even worse, results from this overarching test would be worthless due to the NFL theorem. However, experimental design techniques such as DOE or DACE can be applied to setup experiments efficiently.

Furthermore, it is important that the optimal choice can be selected without wasting disproportional time and effort. This criterion is related to Fredkin's paradox, that will be discussed in Chapter 7. Yet, it is not obvious how many instances of problem P are necessary to demonstrate that algorithm A performs better than algorithm B. Is a test suite with 500 functions more convincing than one with 5 functions?

6.3 Performance Measures

As tuning and comparison of search algorithms can be conducted for many reasons, different performance measures are necessary. Often, the average response value from an algorithm run is optimized. But there are circumstances under which it is desirable to optimize the maximum value and not the average. For example, to guarantee good service for all waiting customers in an elevator system, the maximum waiting time has to be minimized. Otherwise, for some systems, it is more important to minimize the variance in the response than it is to minimize the average value.

Example 6.1 (Mean, median, maximum, and minimum)
Considering the mean function values in Figure 6.1 one might conclude that threshold selection (TS) improves the performance of the (1+1)*-ES especially for high noise levels. Comparing the median values leads to a similar conclusion. And, the comparison of the maximum function values shows that threshold rejection might improve the performance, too.*
However, the situation changes completely if the minimum function values are compared. Surprisingly, no clear difference between the two algorithms can be detected. ■

Obviously, it is not trivial to find adequate performance measures. The performance measure under consideration should lead to a comparison that is well-defined, algorithmic, reproducible, and fair (Johnson, 2002). Dolan and More (2001) discuss several shortcomings of commonly used approaches, i.e. the subjectivity related to the choice of a penalty value that is assigned to algorithms that failed to solve a problem.
We will consider three optimization scenarios before we present measures that refer to efficiency and those that refer to effectivity.

6.3.1 Scenarios

Research testing of new algorithms for existing problems can be distinguished from the development of the most efficient solution procedure for one problem instance (Rardin and Uzsoy, 2001). Eiben and Smith (2003) differentiate between three types of optimization problems: (1) *Design problems* (create one excellent solution at least once), (2) *repetitive problems* (find good solutions for different problem instances), and (3) *on-line control problems* (repetitive problems that have to be solved in real-time).

Following Schwefel (1975, 1995), we will use a classification scheme that distinguishes between effectivity and efficiency. *Effectivity* is related to robustness and deals with the question

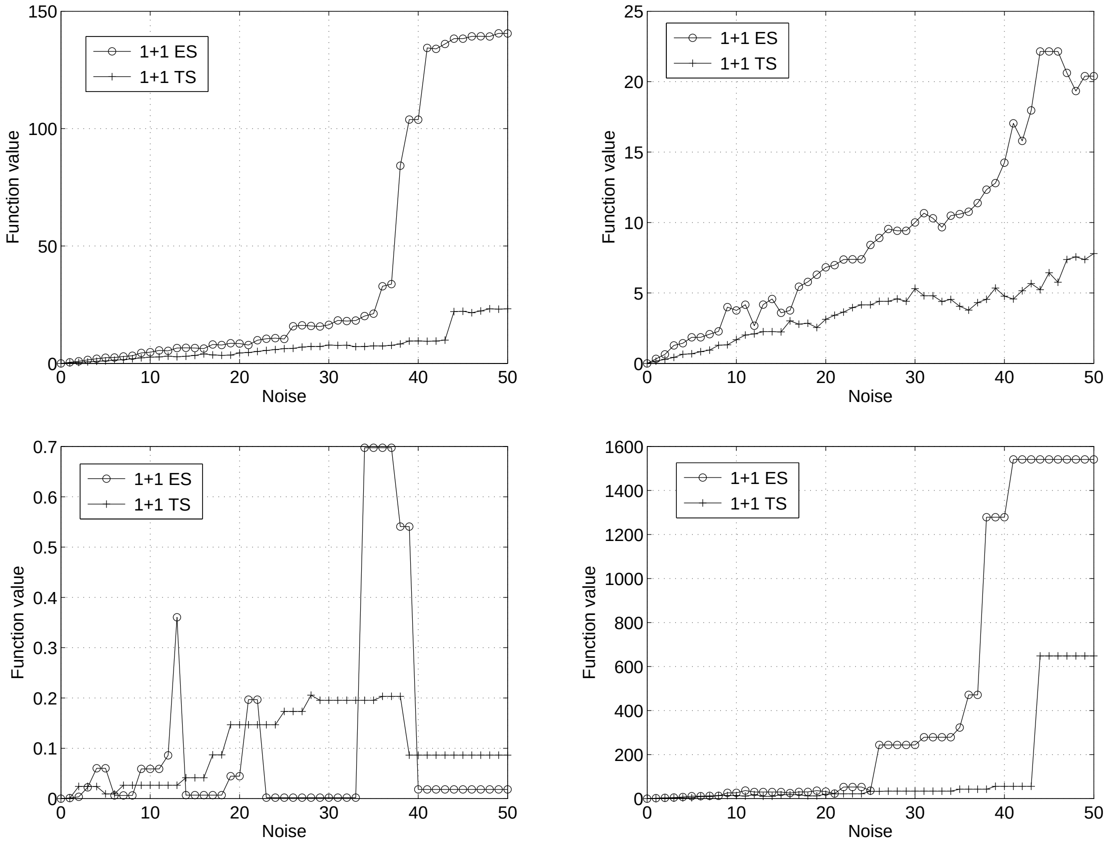


Figure 6.1: *Mean, median, maximum, and minimum function values. An $(1+1)$-ES compared to threshold selection (TS). From left to right: Mean and median (first row), and minimum and maximum function values (second row) from 500 runs for different noise levels.*

whether the algorithm produces the desired effect. On the other hand, the measurement can be based on *efficiency*: Does the algorithm produce the desired effects without waste? The rate of convergence is one typical measure to judge the efficiency of evolutionary algorithms. More et al. (1981) note that many tests do not place enough emphasis on testing the robustness of optimization programs. Most of the testing procedures are focused on their efficiency only. A notable exception in this context is Schwefel (1995), who performed three test series: The first to analyze numerically the rates of convergence for quadratic objective functions, the second to test the reliability of convergence for the general non-linear case, and the third one to investigate the computational effort required for non-quadratic problems.

6.3.2 Effectivity

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. Criteria based on robustness mainly consider the best result. Robustness refers to the hardness or complexity of the problem. Therefore, the analysis can be based on low dimensional problems. Due to the lack of computing resources, Schwefel (1975) considered 50 problems with dimensions from one to six only.

Machine precision demands the specification of a border f_{border} to distinguish solutions that have found a function value sufficiently close to the optimum value from solutions that failed to obtain this value. The machine precision ϵ is the largest positive number that $1 + \epsilon = 1$. For a computer that supports the IEEE standard 754, double-precision ϵ is $2^{-52} \approx 2.224 \cdot 10^{-16}$. A simplified variant of the border determination in Schwefel (1995, p. 206), reads as follows: Let $\epsilon \in \mathbb{R}_+$ be a real valued positive constant, for example the machine precision ϵ. Determine

$$f_{\text{border}} = \begin{cases} \max\{f(x^* + \epsilon x^*), f(x^* - \epsilon x^*)\} & \text{if } x^* \neq 0, \\ \max\{f(\epsilon \mathbf{1}), f(-\epsilon \mathbf{1})\} & \text{otherwise,} \end{cases}$$

where $\mathbf{1}$ denotes the vector of ones: $\mathbf{1} = (1, 1, \ldots, 1)^T$. We will list some commonly used performance measures (PM) to analyze the effectivity of an algorithm in the following.

(PM-1) If the optimal solution is known, the percentage of run configurations terminated successfully, the *success ratio* (SCR), can be used to measure the performance of an algorithm. The success ratio has already been mentioned in Section 2.6 in the context of logistic regression models. Two variants of this measure can be defined, it can be based on the distance of the obtained best objective function value $\tilde{f}$ to the best known function value f^*, or on the distance of the position with the obtained best objective function value $\hat{x}$ to the position of best known function value x^*. Unless otherwise explicitly stated, we will use the variant that measures the distance between $\tilde{f}$ and f^*.

To measure the algorithm's progress towards a solution, one can specify a budget, i.e. the number of function evaluations available to an algorithm. If the starting points were chosen randomly, or if stochastic search algorithms were analyzed, several solutions are obtained for one algorithm configuration. We list only three possible ways to evaluate the results. Other measures, i.e. based on the median, are possible.

(PM-2) Schwefel (1995, p. 211) selects out of n tests the one with the best end result.

(PM-3) The *mean best function value* can be defined as the average value of the best function values found at termination for one specific run configuration. This performance measure will be referred to as MBST.

(PM-4) The best function value found by algorithm A is recorded. By starting A from a number of randomly generated initial points, a sample is obtained. Trosset and Padula (2000) state that the construction of a non-parametric estimate of the probability density function from which the sample was drawn has an "enormous diagnostic value" to study the convergence of iterative algorithms to local solutions.

6.3.3 Efficiency

The methods presented in Section 6.3.2 specify the available resources in advance and ask how close to the optimum an algorithm could come. Diametrically opposed to these methods are those that measure the required resources. They measure the efficiency of an algorithm, for example the number of function evaluations or the timing of the algorithm. This difference is depicted in Figure 6.2.

(PM-5) Considering the quality of the best solution, it is a common practice to show a graph of the solution quality versus time.

(PM-6) 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.

(PM-7) The *run length distribution* (RLD) as introduced in Hoos (1998) provides suitable means to measure performance and to describe the qualitative behavior of optimization algorithms.
A typical run length distribution is shown in Figure 6.3. The algorithm to be analyzed is run n times with different seeds on a given problem instance. The maximum number of function evaluations $t_{\max}$ is set to a relatively high value. For each successful run the number of required function evaluations, t_{run}, is recorded. If the run fails, t_{run} is set to infinity. The empirical cumulative distribution (CDF) is the cumulative distribution that represents these results. Let $t_{\text{run}}(j)$ be the run length for the j-th successful run. Then, the empirical CDF is defined as

$$\Pr\left(t_{\text{run}}(j) \leqslant t\right) = \frac{\{\#j \,|\, t_{\text{run}}(j) \leqslant t\}}{n}, \tag{6.1}$$

where $\{\#j \,|\, t_{\text{run}}(j) \leqslant t\}$ denotes the number of indices j, such that $t_{\text{run}}(j) \leqslant t$. RLDs are based on methods proposed in Parkes and Walser (1996). Exponential RLD can be used to determine whether a restart is advantageous. The exponential RLD is memoryless, because the probability of finding a solution within an interval $[t, t+k]$ does not depend on the actual iteration i. If the RLD is exponential, the number of random restarts does not affect the probability of finding a solution with a given interval. Otherwise, if the RLD is not exponential, there may exist some iteration for which a restart is beneficial. The reader is referred to the discussion in Chiarandini and Stützle (2002).

(PM-8) Efficiency rates measure progress from the starting point $x^{(0)}$ as opposed to convergence rates that use a point in the vicinity of the optimum x^*. Hillstrom (1977) defines

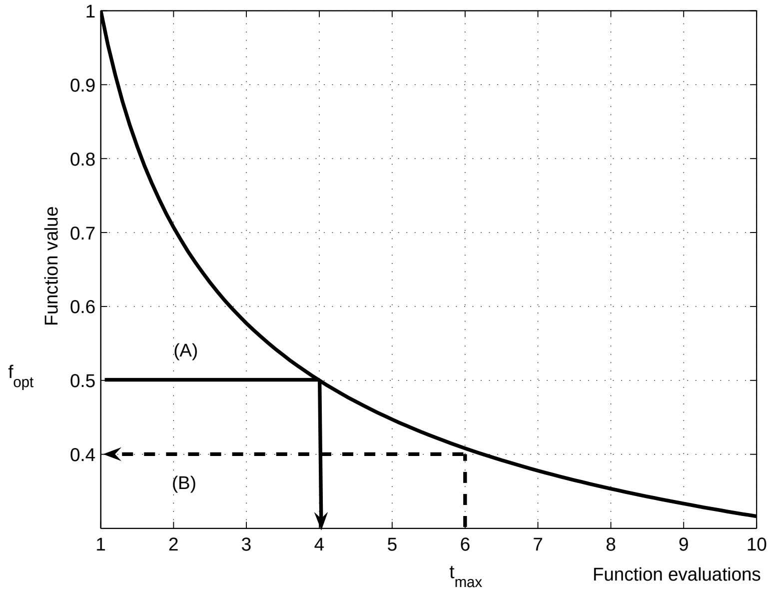


Figure 6.2: *Two diametrically opposed methods to determine the performance of an algorithm. (A) measures the required resources to reach a given goal, (B) measures the obtained function value that can be reached with a prespecified budget.*

the following efficiency measure:

$$MTER = \ln(|f^{(0)} - f^*|/|\hat{f} - f^*|)/T, \tag{6.2}$$

where T is an estimate of the elapsed time in centiseconds, and where $f^{(0)}$, f^*, and $\hat{f}$ are the initial, known, and final minimum values of the objective function. The difference $|\hat{f} - f^*|$ is bounded by the machine precision (Hillstrom, 1977). Thus, Hillstrom's definition is not machine independent.

(PM-9) 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 function value found, and t_{best} is the time to produce that best value (Barr et al., 1995). Run length distributions provide a graphical presentation of this relationship.

Example 6.2
If an algorithm A requires $t^*(A) = 30,000$ function evaluations to produce the best solution $f^*(A) = 10$ and $t_{0.05}(A) = 10,000$ evaluations to produce a solution that is smaller than $f_{0.05}(A) = 10.5$, than its quality-effort relationship is $30000/10000 = 3$. ■

Several measures have been defined for evolutionary algorithms. These measures have been introduced to measure the search progress, and not the convergence properties of an algorithm. For example, the corridor model with objective function

$$f(x) = -\sum_{i=1}^{d} x_i \tag{6.3}$$

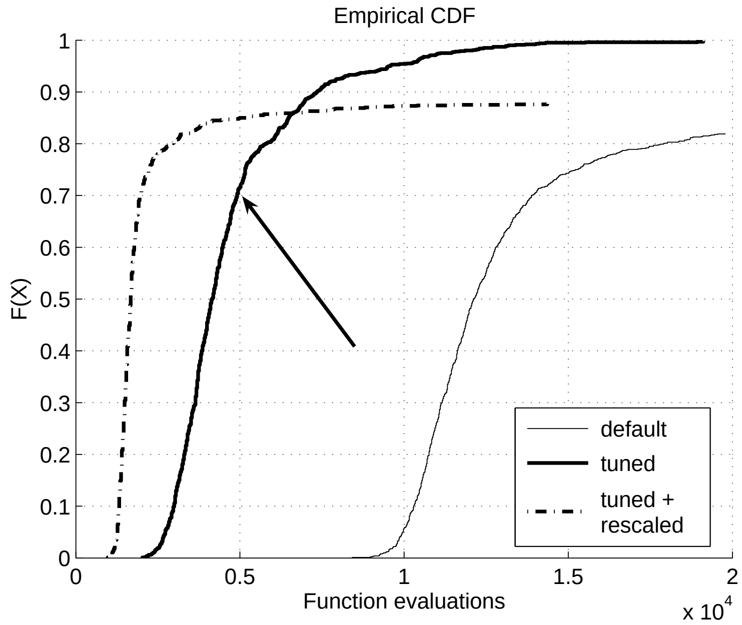


Figure 6.3: *PSO. Run length distributions of the default, the tuned and the tuned and rescaled inertia variant of the particle swarm optimization. The arrow indicates that seventy percent of the runs of the tuned algorithm can detect the optimum with less than 5000 function evaluations. After 15,000 function evaluations nearly every run is able to detect the optimum. The* $w_{iterScale}$ *scheme is modified in the tuned and rescaled variant. It outperforms the other algorithms if the number of function evaluations is low. However, increasing the number of function evaluations from 5000 to 15,000 does not improve the success ratio of the tuned and rescaled PSO. The default configuration was able to complete only 80 percent of the runs after 20,000 function evaluations. RLDs provide good means to determine the maximum number of function evaluations for a comparison.*

and constraints as defined in Schwefel's Problem 3.8 (Schwefel, 1995, p. 364) has its minimum at infinity. It was used to analyze "the cost, not of reaching a given approximation to an objective, but of covering a given distance along the corridor axis" (Schwefel, 1995, p. 365). Especially dynamic environments require algorithms that can follow the moving optimum, convergence is ipso facto impossible. Additionally, these measures consider that the best solution found during the complete run may not belong to the final population, for example in comma-strategies:

(PM-10) 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 g generations $f^{(g)}$:

$$c(g) = \log(\sqrt{f^{(0)}/f^{(g)}}).$$

This measure is related to the efficiency measure $MTER$ defined above in Equation 6.2. Kursawe (1999) uses the normalized convergence velocity:

$$dc(g)/g,$$

where d denotes the problem dimension.

(PM-11) Rudolph (1997) analyzes the first hitting time of an ϵ-environment of the global optimum of the objective function f^*.

(PM-12) Arnold and Beyer (2003) define the efficiency of a search algorithm as the ratio of the expected one-generation gain to the average number of objective function evaluations per generation.

Furthermore, we present three performance measures that are commonly used for evolutionary algorithms:

(PM-13) The quality gain $\overline{Q}$ measures the expected change of the function value from generation g to $g+1$. It is defined as

$$\overline{Q} = E\left[1/s\sum_{i=1}^{s} f_i^{(g+1)} - 1/s\sum_{i=1}^{s} f_i^{(g)}\right], \tag{6.4}$$

where s denotes the population size.

(PM-14) The progress rate φ is a distance measure in the parameter space to measure the expected change in the distance of the parent population to the optimum x^* from generation g to $g+1$. This measure will be referred to as PRATE.

(PM-15) The success rate s_r is the ratio of the number of the successes to the total number of trials, i.e. mutations. It can be used as local performance measure from generation g to $g+1$. Note, the success rate was introduced in Chapter 5.3 to define the 1/5 success rule for evolution strategies.

The reader is referred to Beyer (2001) for a comprehensive discussion of these measures.

The performance measures considered so far are based on one problem instance only. The following measures compare i different problem instances on j different algorithm instances.

(PM-16) For the i-th problem instance and the j-th algorithm, we can define

$$t_{i,j} = \text{time required to solve problem } i \text{ by algorithm } j. \tag{6.5}$$

The distribution function of a performance metric, the *performance profile*, shows important performance characteristics. It can be used to determine the computational effort (Dolan and More, 2001; Bussieck et al., 2003). The *performance ratio* is defined as

$$r_{i,j} = \frac{t_{i,j}}{\min\{t_{i,j} : 1 \leq j \leq n_j\}}, \tag{6.6}$$

where n_j denotes the number of algorithms and $t_{i,j}$ is defined as in Equation 6.5. The performance ratio $r_{i,j}$ compares the performance on problem i by algorithm j with the best performance by any algorithm on this problem. We can define

$$\rho_j(t) = \frac{1}{n_j}\#\{i : r_{i,j} \leq t\}, \tag{6.7}$$

the cumulative distribution function for the performance ratio. The situation from Table 6.2 is depicted in Figure 6.4.

Table 6.1: *Test instances and algorithms. The entries in the ith row and jth column present $t_{i,j}$, the number of function evaluations in units of 10^3 required to solve problem instance i with algorithm j as defined in Equation 6.5.*

	Algorithm 1	Algorithm 2	Algorithm 3
Instance 1	10	12	5
Instance 2	10	13	30
Instance 3	20	40	100

Table 6.2: *Performance ratios ρ_{ij} for the values from Table 6.1*

	Algorithm 1	Algorithm 2	Algorithm 3
Instance 1	2	2.4	1
Instance 2	1	1.3	3
Instance 3	1	2	5

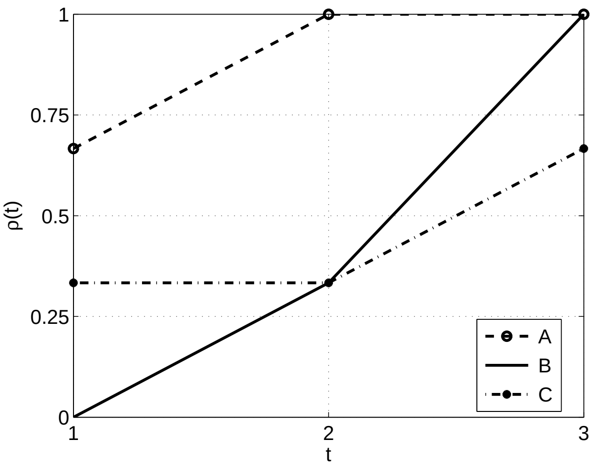


Figure 6.4: *Cumulative distribution function for the performance ratio $\rho_j(t)$. Values taken from Table 6.2. Larger values are better. Algorithm A performs best, whereas the performances of B and C cannot be distinguished.*

(PM-17) A common practice to compare global optimization algorithms (solvers) is to sort the problem instances by the time taken by a reference solver j_0: $t_i := t_{i,(j_0)}$. Then for every solver i the time taken $t_{i,j}$ is plotted against this ordered sequence of problem instances. Instances, for which the optimum was not found by the solver within the allotted time $t_{\max}$ get a dummy time above the timeout value. The successful completion of the optimization task can be assessed directly. How the performance of a solver scales with the problem dimension can only be seen indirectly, since the values refer to the ordering of the problem instances induced by the reference solver (Neumaier et al., 2004).

(PM-18) Schwefel (1995, p. 180) suggests the following procedure to test the theoretical predictions of convergence rates: Test after each generation g, if the interval of uncertainty of the variables has been reduced by at least 90 %:

$$|x_i^{(g)} - x_i^*| \leq \frac{1}{10}|x_i^{(0)} - x_i^*|, \qquad \text{for } i = 1, \ldots, d, \tag{6.8}$$

where x^* denotes the optimum and the values x_i were initialized as in Equation 4.5 in Example 4.4. The number of function calls can be displayed as a function of the numbers of parameters (dimension) on a log-log scale as shown in Figure 6.5.

Since run times depend on the computer system, measures for computational effort might be advantageous: Counting operations, especially for major subtasks such as function calls can explicitly be mentioned in this context. Finally, we can annotate that Barr and Hickman (1993) discuss performance measures for parallel algorithms.

6.3.4 How to Determine the Maximum Number of Iterations

Measures based on effectivity often require the specification of $t_{\max}$ the maximum number of iterations before the run is started. A wrong specification of the $t_{\max}$ value may lead to a bad experimental design. Test problems that are too easy may cause ceiling effects. If algorithms A and B achieve the maximum level of performance (or close to it), the hypothesis "performance(A) $\geq$ performance(B)" should not be confirmed (Cohen, 1995). Floor

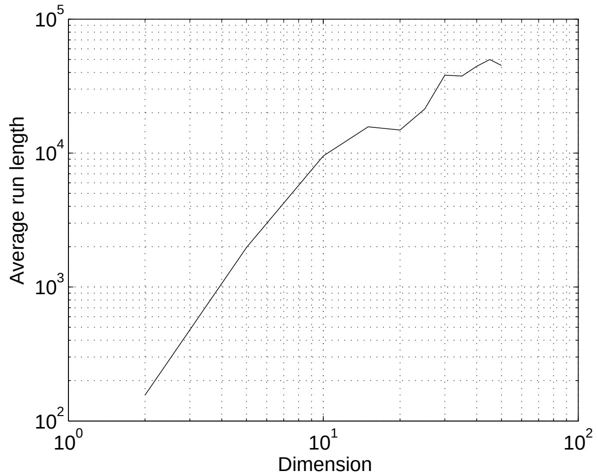


Figure 6.5: *Average run length (avgRunLength) vs. problem dimension. The number of function calls is displayed as a function of the numbers of parameters (dimension) on a log-log scale. A similar presentation was chosen in Schwefel (1995) to test convergence rates for a quadratic test function. The figure depicts data from an analysis of a particle swarm optimization on the sphere function.*

effects describe the same phenomenon on the opposite side of the performance scale: The test problem is too hard, nearly no algorithm can solve it correctly.

Example 6.3 (Floor effects)
If the number of function evaluations is chosen too small, floor effects can occur. Consider Rosenbrock's function, the starting point $x_i^{(0)} = (10^6, 10^6)^T$*, and a budget of 10 function evaluations only.* ■

Example 6.4 (Ceiling effects)
The randomly generated instances of the TSP *as described in objection (OB-1) in Section 3.5 can produce ceiling effects, because any random guess produces a good solution.* ■

Run length distributions as presented in Section 6.3.3 can be used to determine an appropriate value for $t_{\max}$.
Schaffer et al. (1989) propose a technique to determine the total number of iterations $t_{\max}$ and to prevent ceiling effects: The number k belongs to the set $\mathcal{N}$, if at least 10% of the algorithm design configurations $x_a \in \mathcal{D}_A$ located the known best objective function value f^* at least on average every second time after k iterations. The number of function evaluations at which to compare different algorithm designs is chosen as $N_{\text{tot}} = \min\{\mathcal{N}\}$.

Example 6.5
Based on the data from Table 6.1, we can see that algorithm 1 *was able to locate the optimum at least every second time after* $10,000$ *function evaluations, algorithms* 2 *and* 3 *required* $13,000$ *and* $30,000$ *evaluations respectively and* $\mathcal{N} = \{10,000; 13,000; 30,000\}$*. A choice of* $k = 10,000$ *as the minimum number of function evaluations guarantees that at least 10 percent of the algorithms locate the optimum after* k *iterations. At least 10 percent is in our case one algorithm only, because three algorithms are considered.* ■

6.4 The Classical DOE Approach

Rather than detail the classical DOE procedure here, since it is fully outlined in Bartz-Beielstein (2003), we give an overview and make some comments that reflect further experiences that we have observed in the meantime.

6.4.1 A Three-Stage Approach

In classical DOE the following three-stage approach is proposed:

Screening

Consider an algorithm with k exogenous strategy parameters, for example an evolution strategy with $k = 9$ parameters. Screening analyzes the main effects only. Possible interactions will be investigated later. Therefore, we recommend to use fractional-factorial 2^{k-p} designs with $1 \leq p < k$. These are orthogonal designs that require a moderate number of experiments. Due to the orthogonality of these designs, 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 (Box et al., 1978). Roman numerals denote the corresponding design resolution. Our first experiments are based on resolution III designs. These designs ensure that no main effect is confounded with any other main effect, but main effects can be confounded with two-factor interactions. Fractional-factorial 2^{k-p} 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 that will be used during optimization, the third stage of the classical DOE approach.

Modeling

First or second order interactions can be taken into account, because only the important factors that have been detected during the screening phase will be analyzed further. At this stage, resolution IV or resolution V designs are recommended. Half-normal plots can be used to display the main effects and interactions. A linear approximation may be valid in a subdomain of the full experimental area. *Response surface methodology* (RSM) is a collection of mathematical and statistical tools to model, analyze and optimize problems where the response of interest is influenced by several variables (Montgomery, 2001). In RSM, we determine the direction of improvement using the "path of the steepest descent" (minimization problem) based on the estimated first-order model (Kleijnen and Van Groenendaal, 1992). 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. We can repeat this step until the expected optimum area is found (if the the response surface is unimodal). There the linear model is inadequate and shows significant lack-of-fit. We cannot determine a direction of improved response in this case.

Optimization

Central composite designs that can be complemented with additional axial runs are often used at this experimental stage. They can be combined with response surface methods and require a relatively high number of runs. We apply the standard techniques from regression analysis for meta-model validation (Draper and Smith, 1998). 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.

Example 6.6 (2^{3-1} designs)
Consider a design for three factors A, B, and C with two levels each. One-half fraction of the full factorial 2^3 design is called a 2^{3-1} fractional-factorial design. Plus and minus signs can be used to denote high and low factor levels respectively. If we define the multiplication of two factors by their associated levels as $++ = -- =: +$ and $+- = -+ =: -$, then one-half fraction of the design is given by selecting only those combinations with $ABC = +$, for example: $A = -$, $B = -$, and $C = +$. This design is a resolution III design. ■

6.4.2 Tuning an Evolution Strategy

Bartz-Beielstein (2003) describes a situation in which only a few preliminary experiments can be performed to find a suitable ES parameter setting. To start, an experimental region (design space) has to be determined. The design space is defined as

$$I := [a_1, b_1] \times \ldots \times [a_d, b_d] \subseteq \mathbb{R}^d, \tag{6.9}$$

with the center point $z_i = (a_i + b_i)/2, \quad i, \dots, d$, as depicted in Figure 4.1. The optimization response is approximated in the experimental region by the first-order regression model, cf. Equation 2.11. The range of a coded or standardized variable x is bounded by $[-1, 1]$. The range $[a, b]$ of the corresponding original (natural) variable z can be mapped by a linear transformation to $[-1, 1]$.

An ES as presented in Beyer and Schwefel (2002) has at least 9 different exogenous parameters. To model the ES performance, five quantitative variables $(\mu, \nu, \sigma^{(0)}, c_\tau)$ and four qualitative variables $(n_\sigma, r_x, r_\sigma, \kappa)$ have to be considered. The number of offspring λ can be determined from the size of the population μ and value of the selective pressure $\nu = \lambda/\mu$. The inputs μ and ν are treated as quantitative factors, their values are rounded to the nearest whole number to get a set of working parameters. The maximum life span κ is treated as a qualitative factor, because only comma and plus selection schemes have been analyzed.

Instead of using a full factorial 2^k design that would require 512 optimization runs, a 2^{9-5}_{III} fractional-factorial design, that requires only 16 optimization runs, was chosen. Box et al. (1978) give rules for constructing fractional-factorial designs for the ES factors and levels from Table 5.3. Table 6.3 shows the 16 run configurations.

The interval $[-1\text{Level}, +1\text{Level}]$ from Table 6.4 contains values that have been proposed in Bäck (1996). For example, we have chosen the experimental region $I_1 = [10, 20]$ for μ, because it includes the recommended value $\mu = 15$.

A First Look at the Data

Histograms or scatterplots can be used to detect outliers easily. As randomness is replaced by pseudo-randomness, we do not recommend to simply exclude outliers from the analysis. Removing potential outliers may destroy valuable information. Instead, we recommend to look at the raw data that are tabulated and sorted. Specifying a better suited experimental region for factors that arouse suspicion might prevent outliers.

Example 6.7 (Outliers and Experimental Region)
We can conclude from Table 6.5 that the choice of a value of 20 *as the second level for factor A should be reconsidered.* ■

Regression Analysis

Regression analysis and stepwise model selection by *Akaike's information criterion* (AIC) have been performed for the coded variables x_i. To perform the experiments, these values have be re-transformed to the natural variables z_i. Before we start the search along the path of the steepest descent, the adequacy of the regression model is tested, and a check for interactions is performed. Regression analysis reveals that only three of the nine factors are important: (1) The initial sigma value $\sigma^{(0)}$, (2) the population size μ, (3) and the selective pressure ν. Plus and comma strategies are tested in parallel, because they perform similar at this stage of experimentation.

Starting from the center point we perform a line search in the direction of the steepest descent that is given by

$$-(\hat{\beta}_1, \dots, \hat{\beta}_k).$$

To determine the step-sizes Δx_i for the line search, 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

Table 6.3: *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

Table 6.4: *Evolution strategy: Symbols and levels. Values chosen with respect to the default settings from Table 5.3.*

Symbol	Parameter	Variable	-1	$+1$	Type
μ	Number of parent individuals	x_1	10	20	quant.
ν	Offspring-parent ratio	x_2	5	10	quant.
$\sigma^{(0)}$	Initial standard deviations	x_3	1	5	quant.
n_σ	Number of standard deviations	x_4	1	12	qual.
c_τ	Multiplier for mutation parameters	x_5	1	2	quant.
ρ	Mixing number	x_6	b	m	qual.
r_x	Recombination operator for object variables	x_7	i	d	qual.
r_σ	Recombination operator for strategy variables	x_8	i	d	qual.
κ	Maximum life span	x_9	-1	1	qual.

Table 6.5: Tabulated raw data. The function value y is shown in the first column. Factor A produces outliers, if its high level is chosen.

y	Factor A	Factor B	Factor C	...
0.5	5	5	1	...
0.6	5	10	1	...
0.61	5	5	5	...
0.9	5	10	5	...
⋮	⋮	⋮	⋮	...
259.2	20	10	1	...
277.1	20	5	5	...
297.3	20	5	1	...
433.6	20	10	5	...

is

$$\Delta x_i = -\hat{\beta}_i/(|\hat{\beta}_j|/\Delta x_j), \qquad i = 1, 2, \ldots, k;\ i \neq j.$$

The corresponding numerical values are shown in Table 6.6. The qualitative factors have to be treated separately, because a line search cannot be performed for qualitative factors such as the recombination operator. 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. Before the initial sigma value $\sigma^{(0)}$ reaches the boundaries of the feasible region, the search is stopped. This value may lie outside the experimental region, but has to be a feasible value for the algorithm. A second algorithm design based on the improved setting from the line search was created. The regression analysis shows that the plus selection scheme is advantageous in this case. Therefore, the experiments with the comma strategy, which have been run in parallel, are stopped. Effects caused by the parameters μ, ν, and $\sigma^{(0)}$ are statistically significant and have to be considered further. As only three of the nine parameters remain, a more complex design has been chosen.

Table 6.6: Steepest descent. 12 dimensional sphere function. The line search is stopped after 7 steps to avoid negative $\sigma^{(0)}$ values. Source: Bartz-Beielstein (2003).

Steps Δ	$\sigma^{(0)}$ Coded x_1	ν Coded x_2	μ Coded x_3	$\sigma^{(0)}$ Original z_1	ν Original z_2	μ Original z_3	Mean Response $\log(y)$	Median Response $\log(y)$
	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

Central Composite Designs

A central composite design 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 Figure 4.1. We can conclude from the regression analysis that was based on a 2^3 central composite design that a further decrease of the population size and of the selective pressure might be beneficial. 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 (not shown here). Instead of performing a second line search, we use response surface methods to visualize the region of the local optimum. Data generated from a CCD with axial runs can be used to generate a surface plot. A numerical comparison of the function values obtained from the first and the improved algorithm design reveals a significant improvement, see Table 6.7. We have found a better algorithm design, x^*_{ES}, that improves the performance of the "standard" ES presented in Bäck (1996) for this specific problem. 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. The result found so far does not justify the conclusion that this design is optimal. Our intention is to give the optimization practitioner a framework on how to set up algorithms with working parameter configurations. Further optimization of this setting is possible, but this is beyond the intention of this study. As already noted in Section 4.7 the assumption of a linear model for the analysis of computer algorithms is highly speculative. The applicability of methods from computational statistics, that are not restricted to this assumption, is analyzed in the following section.

6.5 Design and Analysis of Computer Experiments

The term *computational statistics* subsumes computationally intensive methods (Gentle et al., 2004b). Statistical methods, such as experimental design techniques and regression analysis can be used to analyze the experimental setting of algorithms on specific test problems. One important goal in the analysis of search algorithms is to find variables that have a significant influence on the algorithm's performance. Performance measures have been discussed in Section 6.3, i.e. performance can be quantitatively defined as the average obtained function value in a number (e.g. 50) of independent experiments. This measure was also used in Shi and Eberhart (1999). Questions like "How does a variation of the swarm size influence the algorithm's performance?" or "Are there any interactions between swarm size and the value of the inertia weight?" are important research questions that provide an understanding of the fundamental principles of stochastic search algorithms such as PSO.

The approach presented in this section combines classical techniques from statistical de-

Table 6.7: *Evolution strategy. Comparison of the function values from the first and the improved algorithm design.*

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
$x^{(0)}_{\mathrm{ES}}$	0.01	0.04	0.05	0.07	0.09	0.20
x^*_{ES}	$5.32e-64$	$1.16e-59$	$1.25e-57$	$6.47e-51$	$7.54e-56$	$5.17e-49$

sign of experiments (DOE), classification and regression trees (CART), and modern design and analysis of computer experiments (DACE) techniques. Standard text books for DOE, CART, and DACE are Montgomery (2001), Breiman et al. (1984), and Santner et al. (2003) respectively. Bartz-Beielstein and Markon (2004) provide a comparison of DOE, CART and DACE for direct search algorithms.

Since DACE was introduced for deterministic computer experiments, repeated runs are necessary to apply this technique to stochastic search algorithms.

In the following, the specification of the DACE process model that will be used later to analyze our experiments is described. This specification is similar to the selection of a linear or quadratic regression model in classical regression. DACE provides methods to predict unknown values of a stochastic process and it can be applied to interpolate observations from computationally expensive simulations. Furthermore it enables the estimation of the prediction error of an untried point, or the mean squared error (MSE) of the predictor.

6.5.1 Sequential Designs Based on DACE

Prior to the execution of experiments with an algorithm, the experimenter has to specify suitable parameter settings for the algorithm, i.e., a design point x_a from an algorithm design $\mathcal{D}_A$.

Often, designs that use sequential sampling are more efficient than designs with fixed sample sizes. First, an initial design $\mathcal{D}_A^{(0)}$ is specified. Information obtained in the first runs can be used for the determination of the second design $\mathcal{D}_A^{(1)}$ in order to choose new design points more efficiently.

Sequential sampling approaches with adaptation have been proposed for DACE. For example, in Sacks et al. (1989) sequential sampling approaches with and without adaptation were classified to the existing meta-model. We will present a sequential approach that is based on the expected improvement, see Equation 4.4 and Figure 4.5. In Santner et al. (2003, p. 178) a heuristic algorithm for unconstrained global minimization problems is presented. Consider one problem design $\mathcal{D}_P$. Let $y^k_{\min}$ denote the smallest known minimum value after k runs of the algorithm, $y(x)$ be the algorithm's response, i.e. the realization of $Y(x)$ in Equation (2.15), and let x_a represent a specific design point from the algorithm design $\mathcal{D}_A$. Then the improvement is defined as in Equation 4.4

$$\text{improvement at } x_a = y^k_{\min} - y(x_a), \text{ if } y^k_{\min} - y(x_a) > 0, \text{ and } 0, \text{ otherwise.}$$

6.5.2 Sequential Parameter Optimization

The *sequential parameter optimization* (SPO) method, which is developed in this section, describes an implementable but heuristic method. It consists of the twelve steps that are reported in Table 6.8. During the pre-experimental planning phase (S-1) the experimenter defines exactly what is to be studied and how the data are to be collected. The recognition and statement of the problem seems to be a rather obvious task. However, in practice, it is not simple to formulate a generally accepted goal. *Discovery*, *confirmation*, and *robustness* are only three possible scientific goals of an experiment. Discovery asks what happens if new operators are implemented. Confirmation analyzes how the algorithm behaves on different problems, and robustness asks for conditions that decrease the algorithm's performance. Furthermore, the experimenter should take the boundary conditions discussed in Section 6.2 into

Table 6.8: *Sequential parameter optimization (SPO). This approach combines methods from computational statistics and exploratory data analysis to improve (tune) the performance of direct search algorithms. It can be seen as an extension of the guidelines from experimental algorithmics presented in Chapter 1.*

Step	Action
(S-1)	Pre–experimental planning
(S-2)	Scientific claim
(S-3)	Statistical hypothesis
(S-4)	Specification of the
	(a) optimization problem
	(b) constraints
	(c) initialization method
	(d) termination method
	(e) algorithm (important factors)
	(f) initial experimental design
	(g) performance measure
(S-5)	Experimentation
(S-6)	Statistical modeling of data and prediction
(S-7)	Evaluation and visualization
(S-8)	Optimization
(S-9)	Termination: If the obtained solution is good enough, or the maximum number of iterations has been reached, go to step (S-11)
(S-10)	Design update and go to step (S-5)
(S-11)	Rejection/acceptance of the statistical hypothesis
(S-12)	Objective interpretation of the results from step (S-11)

account. Statistical methods like run length distributions provide suitable means to measure the performance and describe the qualitative behavior of optimization algorithms.

In step (S-2), the experimental goal should be formulated as a scientific claim, e.g. "Algorithm A, which uses a swarmsize s, which is proportional to the problem dimension d outperforms algorithms that use a constant swarmsize."

A statistical hypothesis, such as "There is no difference in means comparing the performance of the two competing algorithms", is formulated in the step (S-3) that follows.

Step (S-4) requires at least the specification of

(a) an optimization problem,

(b) constraints (for example the maximum number of function evaluations),

(c) an initialization method,

(d) a termination method,

(e) an algorithm, and its important factors,

(f) an initial experimental design, and

(g) a measure to judge the performance.

Regarding (c), several methods have been used for the initialization of the population in population-based algorithms, or the determination of an initial point, $x^{(0)}$, in algorithms that use a single search point. For example, an asymmetric initialization scheme was used in Shi and Eberhart (1999), where the initial positions of the particles, $x_i^{(0)}$, $i = 1, \dots, s$, were chosen uniformly distributed in the range $[15, 30]^d$. Initialization method DETMOD, that uses deterministically modified starting values, was proposed in More et al. (1981).
An algorithm terminates if the problem was solved (XSOL), the algorithm has stalled (STAL), or the resources, e.g. the maximum number of function evaluations, $t_{\max}$, are exhausted (EXH). Note that initialization and termination methods have been discussed in Section 4.6.

The corresponding problem design, $\mathcal{D}_P$, that summarizes the information from (a) to (d) for our experiments with PSO is reported in Table 6.9, while the algorithm design $\mathcal{D}_A$, which represents (e), is reported in Table 6.10 . The experimental goal of the sequential approach presented here can be characterized as the determination of an optimal (improved) algorithm design, x^*_{PSO} for a given problem design $x^{(0)}_{\text{PSO}}$.

At each stage, Latin Hypercube Designs are used. Aslett et al. (1998) report that experience with the stochastic process model had indicated that 10 times the expected number of algorithm design variables is often an adequate number of runs for the initial LHD.

Table 6.9: *Problem design ($\mathcal{D}_P$) for the experiments performed in this chapter. The experiment's name, the number of runs n, the maximum number of function evaluations $t_{\max}$, the problem's dimension d, the initialization method, the termination criterion, the lower and upper bounds, x_l and x_u respectively, for the initialization of the object variables $x_i^{(0)}$, as well as the optimization problem and the performance measure (PM) are reported.*

Design	n	$t_{\max}$	d	Init.	Term.	x_l	x_u	PM
$x^{(1)}_{\texttt{rosen}}$	50	2500	10	NUNIRN	EXH	15	30	MBST

Table 6.10: *PSO: Algorithm designs ($\mathcal{D}_A$) for the inertia weight PSO variant. They correspond to the experiment $x^{(1)}_{\texttt{rosen}}$ of Table 6.9, that optimizes the 10 dimensional Rosenbrock function. $x^{(l)}_{PSO}$ and $x^{(u)}_{PSO}$ denote the lower and upper bounds to generate the LHD, respectively, and x^{*}_{PSO} denotes the parameter settings of the improved design that was found by the sequential approach.*

Design	s	c_1	c_2	$w_{\max}$	w_{scale}	$w_{\text{iterScale}}$	$v_{\max}$
$x^{(l)}_{\text{PSO}}$	5	1.0	1.0	0.7	0.2	0.5	10
$x^{(u)}_{\text{PSO}}$	100	2.5	2.5	0.99	0.5	1	750
x^{*}_{PSO}	21	2.25413	1.74587	0.788797	0.282645	0.937293	11.0496

Example 6.8 (LHD for the PSO constriction variant)
The constriction factor variant of PSO requires the determination of four exogenous strategy parameters, namely the swarm size s, constriction factor χ, parameter $\varphi = c_1 + c_2$, and the maximum velocity $v_{\max}$. Thus, an LHD with at least $m = 15$ design points was chosen. This is the minimum number of design points to fit a DACE *model that consists of a second order polynomial regression model and a gaussian correlation function. The former requires $1 + \sum_{i=1}^{4} i = 11$ design points, while the latter requires 4 design points. Note that for $m = 15$ there are no degrees of freedom left to estimate the mean squared error of the predictor (Santner et al., 2003).* ■

After that, the experiment is run (S-5). Preliminary (pilot) runs can give a rough estimate of the experimental error, run times, and the consistency of the experimental design. Again, RLDs can be very useful. Since we consider probabilistic search algorithms in our investigation, design points must be evaluated several times.

The experimental results provide the base for modeling and prediction in step (S-6). The model is fitted and a predictor is obtained for each response.

The model is evaluated in step (S-7). Several visualization techniques can be applied. Simple graphical methods from exploratory data analysis are often helpful. Histograms and scatterplots can be used to detect outliers. If the initial ranges for the designs were chosen improperly (e.g., very wide initial ranges), visualization of the predictor can guide the choice of more suitable (narrower) ranges in the next stage. Several techniques to assess the validity of the model have been proposed.

Additional graphical methods can be used to visualize the effects of factors and their interactions on the predictors. The 3-dimensional visualizations depicted in Figure 6.7, produced with the DACE toolbox (Lophaven et al., 2002b), have proved to be very useful. The predicted values can be plotted to support the numerical analysis, and the MSE of prediction is used to asses its accuracy. We explicitly note here, that statistical models can provide only guidelines for further experiments. They do not prove that a factor has a particular effect.

If the predicted values are not accurate, the experimental setup has to be reconsidered. This comprehends especially the specification of the scientific goal and the ranges of the design variables. Otherwise, new design points in promising subregions of the search space can be determined (S-8) if further experiments are necessary.

Thus, a termination criterion has to be tested (S-9). If it is not fulfilled, based on the expected improvement defined in Equation 4.4 new candidate design points can be generated (S-10). A new design point is selected if there is a high probability that the predicted output is below the current observed minimum and/or there is a large uncertainty in the predicted output. Otherwise, if the termination criterion is true, and the obtained solution is good enough, the final statistical evaluation (S-11) that summarizes the results is performed. A comparison between the first and the improved configuration should be performed. Techniques from exploratory data analysis can complement the analysis at this stage. Besides an investigation of the numerical values, such as mean, median, minimum, maximum, and standard deviation, graphical presentations such as boxplots, histograms, and RLDs can be used to support the final statistical decision (e.g. see Figure 6.9).

Finally, we have to decide whether the result is scientifically important (S-12), since the difference, although statistically significant, can be scientifically meaningless. As discussed in Section 1.6, an objective interpretation of rejecting or accepting the hypothesis from (S-2) should be presented here. Consequences that arise from this decision are discussed as well. The experimenter's skill plays an important role at this stage. The experimental setup should be reconsidered at this stage and questions like "Have suitable test functions or performance measures been chosen?" or "Did floor or ceiling effects occur?" must be answered. Test problems that are too easy may cause such ceiling effects, cf. the discussion in Section 6.3.4.

6.5.3 Experimental Results

Initially, we investigated Rosenbrock's function. This is a simple and well-known test function to gain an intuition regarding the functioning of the proposed technique. In the next step of our analysis, the S-ring model was considered. We provide a demonstration of the sequential approach by conducting a brief investigation for the Rosenbrock function, using the two variants of PSO as well as the Nelder-Mead simplex algorithm.

Experimental designs and results of PSO or evolutionary algorithms presented in empirical studies are sometimes based on a huge number of function evaluations ($t_{\max} > 10^5$), even for simple test functions. Our goal is to demonstrate how statistical design methods, e.g. DACE, can reduce this number significantly. The proposed approach is thoroughly analyzed for the inertia weight variant of PSO.

6.5.4 Example: Optimizing the Inertia Weight Variant of PSO

This example describes in detail how to tune the exogenous parameters of PSO. It extends the approach presented in Bartz-Beielstein et al. (2004b). Experimental designs and results presented in Shi and Eberhart (1999) have been chosen as a starting point for our analysis.

(S-1) *Pre-experimental planning:* Pre-experimental tests to explore the optimization potential supported the assumption that tuning might improve the algorithm's performance. RLD revealed that there exists a configuration that was able to complete the run successfully using less than 8000 function evaluations, for nearly 80% of the cases. This was less than half the number of function evaluations used in the reference study, justifying the usefulness of the analysis.

(S-2) *Scientific claim:* There exists a parameterization (design point $x^*_{\text{PSO}} \in \mathcal{D}_A$) of PSO that improves its performance significantly for one given optimization problem $x_p \in \mathcal{D}_P$.

(S-3) *Statistical hypothesis:* PSO with the parameterization x^*_{PSO} outperforms PSO with the default parameterization $x^{(0)}_{\text{PSO}}$, which is used in Shi and Eberhart (1999).

(S-4) *Specification:* The generalized 10-dimensional Rosenbrock function, as defined in Equation 3.2, was used for our experiments.

Following the experimental design in Shi and Eberhart (1999), we recorded $\tilde{f}^{(50)}$, the mean function value of the best particle of the swarm at each one of the 50 runs. For the generation of RLD plots, a threshold value to distinguish successful from unsuccessful runs was specified. A run configuration was classified as successful, if $\tilde{f}^{(50)} < \tilde{f}^{(\mathrm{Shi})}$, where $\tilde{f}^{(\mathrm{Shi})} = 96.1715$ is the value reported in Shi and Eberhart (1999). The problem design is shown in Table 6.9. The corresponding setting for the algorithm design is reported in Table 6.10. An algorithm design that covers a wide range of interesting parameter settings (design space) was chosen, and no problem-specific knowledge for the Rosenbrock function was used to perform the experiments, expecting that the sequential approach will guide the search into successful regions.

(S-5) *Experimentation:* Table 6.11 presents numerical results from the optimization process. Each line in Table 6.11 corresponds to one optimization step in the sequential approach. At each step, two new design points are generated and the best one is re-evaluated. This is similar to the selection procedure in $(1+2)$-Evolution Strategies. The number of repeat runs n of the algorithm design points is increased (doubled), if a design has performed best two or more times. A starting value of $n = 2$ was chosen. For example, design point 14 performs best at iteration 1 and iteration 3. It has been evaluated 4 times, therefore the number of evaluations is set to 4 for every newly generated design. This provides a fair comparison and reduces the risk of incorrectly selecting a worse design.

(S-6) *Statistical modeling and prediction:* Following Santner et al. (2003), the response is modeled as a realization of a regression model and a random process as described in Equation (2.15). A gaussian correlation function as defined in Equation (2.16) and a regression model with polynomial of order 2 have been used. Hence, the model reads

$$Y(x) = \sum_{j=1}^{p} \beta_j f_j(x) + Z(x), \tag{6.10}$$

where $Z(\cdot)$ is a random process with mean zero and covariance $V(\omega, x) = \sigma^2 \mathcal{R}(\theta, \omega, x)$. The correlation function was chosen as

$$\mathcal{R}(\theta, \omega, z) \prod_{j=1}^{d} \exp\left(-\theta_j(\omega_j - x_j)^2\right). \tag{6.11}$$

Additionally, at certain stages a tree-based regression model as shown in Figure 6.6 was constructed to determine parameter settings that produce outliers.

(S-7) *Evaluation and visualization:* The MSE and the predicted values can be plotted to support the numerical analysis (we produced all 3-dimensional visualizations with the DACE toolbox (Lophaven et al., 2002b)). For example, the interaction between c_1 and c_2 is shown in Figure 6.7. Values of c_1 and c_2 with $c_1 + c_2 > 4$ generated outliers that might disturb the analysis. To reduce the effects of these outliers, a design correction method has been implemented, namely $c_1 = c_2 - 4$, if, $c_1 + c_2 > 4$. The right part of Figure 6.7 illustrates the estimated MSE. Since no design point has been placed in the

Table 6.11: *Problem design $x^{(1)}_{\texttt{rosen}}$. Inertia weight PSO optimizing the 10-dimensional Rosenbrock function. Each row represents the best algorithm design at the corresponding tuning stage. Note, that function values (reported in the first column) can worsen (increase) although the design is improved. This happens due to the noise in the results, y. The probability that a seemingly good function value that is in fact worse, might occur, decreases during the sequential procedure, because the number of re-evaluations is increased. The number of repeats, n, is doubled if a configuration performs best twice. The corresponding configurations are marked with an asterisk.*

y	s	c_1	c_2	$w_{\max}$	w_{Scale}	$w_{\text{IterScale}}$	$v_{\max}$	Config
6.61557	26	1.45747	1.98825	0.712714	0.481718	0.683856	477.874	14
18.0596	39	1.30243	1.84294	0.871251	0.273433	0.830638	289.922	19
71.4024	26	1.45747	1.98825	0.712714	0.481718	0.683856	477.874	14*
78.0477	30	2.21960	1.26311	0.944276	0.289710	0.893788	237.343	3
75.6154	30	2.21960	1.26311	0.944276	0.289710	0.893788	237.343	3*
91.0935	18	1.84229	1.69903	0.958500	0.256979	0.849372	95.1392	35
91.5438	21	1.05527	1.25077	0.937259	0.498268	0.592607	681.092	43
93.7541	11	1.58098	2.41902	0.728502	0.469607	0.545451	98.9274	52
93.9967	93	1.71206	1.02081	0.966302	0.378612	0.972556	11.7651	20
99.4085	39	1.30243	1.84294	0.871251	0.273433	0.830638	289.922	19*
117.595	11	1.13995	2.31611	0.785223	0.236658	0.962161	56.9096	57
146.047	12	1.51468	2.48532	0.876156	0.392995	0.991074	261.561	1
147.410	22	1.72657	2.27343	0.710925	0.235521	0.574491	50.5121	54
98.3663	22	1.72657	2.27343	0.710925	0.235521	0.574491	50.5121	54*
41.3997	21	2.25413	1.74587	0.788797	0.282645	0.937293	11.0496	67*
43.2249	21	2.25413	1.74587	0.788797	0.282645	0.937293	11.0496	67*
53.3545	21	2.25413	1.74587	0.788797	0.282645	0.937293	11.0496	67*

ranges $1 < c_1 < 1.25$ and $2.25 < c_2 < 2.5$, the MSE is relatively high. This might be an interesting region where a new design point will be placed during the next iteration. Figure 6.8 depicts the same situation as Figure 6.7 after the determination of the design correction. In this case, a high MSE is associated with the region $c_1 + c_2 > 4$, but no design point will be placed there.

(S-8) *Optimization:* Termination or design update. Based on the expected improvement defined in Equation (4.4), two new design points $x_{\text{PSO}}^{(1)}$ and $x_{\text{PSO}}^{(2)}$ are generated. These two designs are evaluated and their performances are compared to the performance of the current best design. The best design found so far is re-evaluated. The iteration terminates, if a design was evaluated for $n = 50$ times and the solution is obtained. The values in the final model read $s = 21$, $c_1 = 2.25$, $c_2 = 1.75$, $w_{\max} = 0.79$, $w_{\text{scale}} = 0.28$, $w_{\text{iterScale}} = 0.94$, and $v_{\max} = 11.05$. This result is shown in the last row of Table 6.11.

(S-11) *Rejection or acceptance:* Finally, we compare the configuration from Shi and Eberhart (1999) to the optimized configuration. The final (tuned) and the first configurations are repeated 50 times. Note, Shi and Eberhart (1999) coupled $x_{\max}$ with $v_{\max}$ as described in Equation 5.5. The mean function value was reduced from 1.84×10^3 to 39.70, the median from 592.13 to 9.44, the standard deviation decreases from 3.10×10^3 to 55.38. Minimum and maximum function values from 50 runs are smaller (64.64 to 0.79 and 18519 to 254.19, respectively). Histograms and boxplots are illustrated in Figure 6.9 for both variants of PSO. The tuned design of the inertia weight PSO variant clearly improves the performance of the PSO algorithm. The statistical analysis from this and from further experiments is reported in Table 6.13. Performing a classical t-test indicates that the null hypothesis "There is no difference in the mean performances of the two algorithms" can be rejected at the 5% level.

(S-12) *Objective interpretation:* The classical tools from exploratory data analysis such as boxplots or histograms indicate that the tuned PSO version performs better than the default PSO, cf. Step (S-11). An NPT t-test comes to the same conclusion. So far we have applied methods from classical and modern statistics to test statistical hypotheses. The results indicate that the null hypothesis H should be rejected. The next step of our analysis describes how we can interpret this rejection in an objective manner and how the relationship between statistical significance and scientific import as depicted in Figure 1.2 can be made more understandable. But before a statistical analysis is performed, we recommend to look at the raw data. Is the obtained result plausible? A comparison of the default design $x_{\text{PSO}}^{(0)}$ to the improved design x_{PSO}^{*} reveals that smaller swarm size s and $v_{\max}$ values improve the algorithm's performance for the problem design under consideration. The value of the cognitive parameter, c_1, should be increased, whereas the value of the social parameter, c_2, should be reduced. The parameters related to the scaling of the inertia weight, w, should be reduced, too. The improved design does not contain any exceptional parameter settings. It appears to be a reasonable choice. Methods that answer the question "*Why* does an increased value of c_1 lead to a better performance?" will be discussed in Chapter 7. Here, we are interested in the question "Does algorithm A outperform algorithm B?"
But how can we be sure that the related PSO performance is better than the performance of the default PSO? To tackle this problem error statistical tools can be used. How do

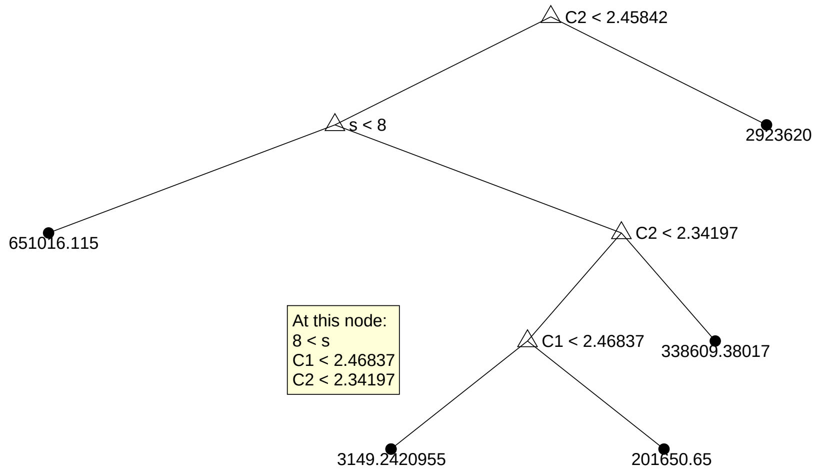


Figure 6.6: *Regression tree. Values at the nodes show the average function values for the associated node. The value in the root node is the overall mean. The left son of each node contains the configurations that fulfill the condition in the node.* $c_1 + c_2 > 4$ *produce outliers that complicate the analysis. In addition, this analysis shows that the swarm size* s *should be larger than* 8.

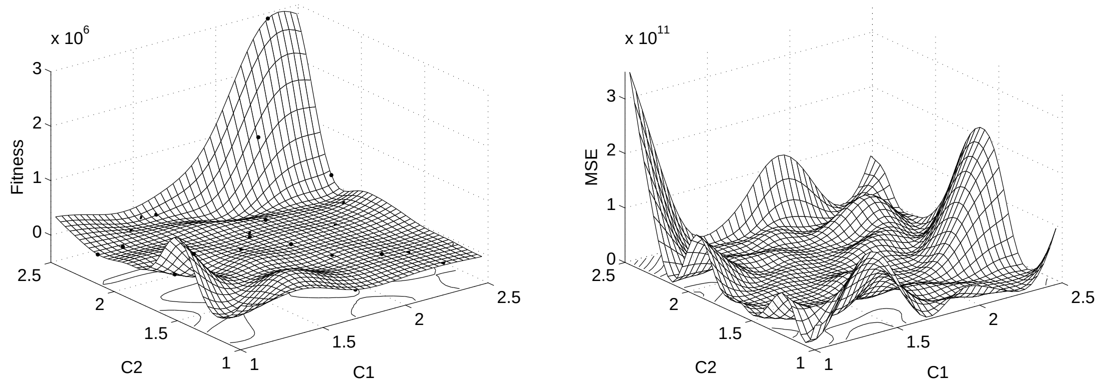


Figure 6.7: *Predicted values (left) and MSE (right). As can be seen in the left figure,* $c_1 + c_2 > 4$ *produce outliers that complicate the analysis. The plots present results from the same data as the regression tree in Figure 6.6.*

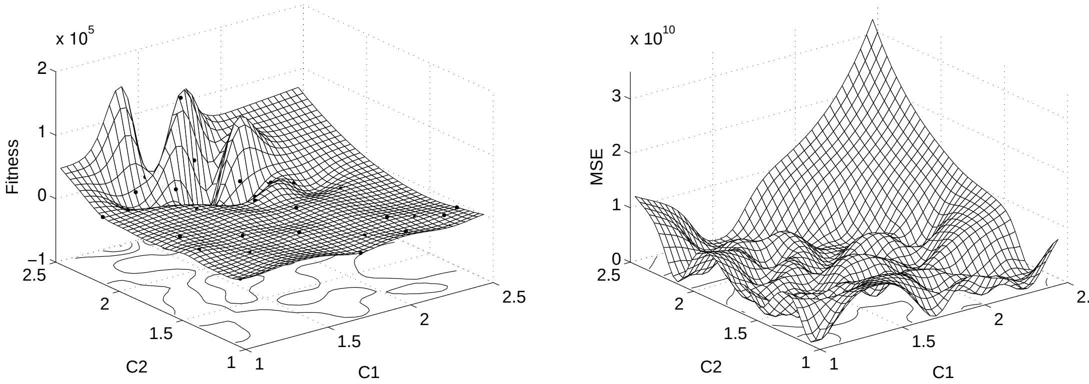


Figure 6.8: *Predicted values (left) and MSE (right). The design correction avoids settings with* $c_1 + c_2 > 4$ *that produce outliers (left). Therefore, a high mean squared error exists in the excluded region (right).*

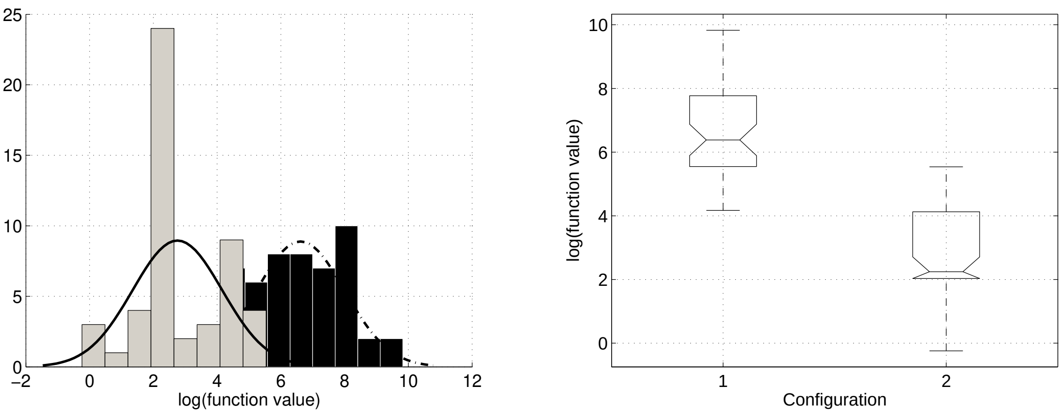


Figure 6.9: *Histogram and boxplot. Left: Solid lines and light bars represent the improved design. Right: The default configuration is denoted as 1, whereas 2 denotes the improved variant. Both plots indicate that the tuned inertia weight PSO version performs better than the default version.*

NPT* interpretations go beyond the results found with NPT tools? This question is closely related to the problem stated in Example 1.1 at the beginning of this thesis.

We claim that statistical tests are means of learning. We are interested in detecting discrepancies between the correct model and a hypothesized one. Experiments provide means to observe the difference between a sample statistic and a hypothesized population parameter. The distribution of the test statistic S can be used to control error probabilities. A rejection of a statistical hypothesis H can be misconstrued if it is erroneously taken as an indication that a discrepancy of scientific importance has been detected. Plots of the observed significance as introduced in Section 1.6 are valuable tools that can be used to detect whether this misconstrual occurs and to evaluate the scientific meaning.

We will consider the case of rejecting a hypothesis H first. An NPT* interpretation of accepting a hypothesis will be discussed later on.

The relationship between the observed significance level $\alpha_{\overline{d}}(\delta)$, the difference in means $\delta = \mu_1 - \mu_2$ of the default PSO, and the tuned PSO version is illustrated in Figure 6.10. First consider the values of the observed significance level $(\alpha_{\overline{d}}(\delta))$ for a sample size of $n = 50$, where $\overline{d}$ denotes the observed difference in means and δ the hypothesized difference as introduced in Section 1.6. The observed difference in means is $\overline{d} = 1798.6$, one standard deviation unit has the value $\sigma_{\hat{x}} = 410.84$. A t-test indicates that the null hypothesis can be rejected at the 5% level. Occurs the observed difference in means due to the experimental error only and is the rejection of the null hypothesis H misconstrued? A difference in means of less than 1 or 2 standard error units might be caused by experimental errors. Observe the values of $\alpha(1798.6, \delta) = \alpha_{1798.6}(\delta)$. How often does a rejection arise when various populations are observed? Since an observed significance level $\alpha_{1798.6}(410.84) = 0.01$ is small, it is a good indication that we are observing two populations with a difference in means $\delta > 410.84$. If one observes a difference $\overline{d}$ when the true difference in means δ was no greater than 410.84, only 1% of the observed differences would be this large. This gives good reasons to reject the associated null hypothesis $H : \delta \leq 410.84$. And, we can learn even more from this result: It also is an indication that $\delta > 822$, since $\alpha_{1798.6}(828) \approx 0.05$. The situation depicted in Figure 6.10 is similar to the situation discussed in Example 1.5. Low $\alpha_{\overline{d}}(\delta)$ values are not due to large sample sizes only. Therefore the statistical results indicate that there is a difference in means and this difference is also scientifically meaningful.

6.5.5 Optimizing the PSO Constriction Factor Variant

The design of the PSO constriction factor variant was tuned in a similar manner as the inertia weight variant. The initial LHD is reported in Table 6.12, where $x^{(l)}_{\text{PSOC}}$ and $x^{(u)}_{\text{PSOC}}$ denote the lower and upper bounds of the experimental region, respectively x^{*}_{PSOC} is the improved design that was found by the sequential procedure, and $x^{(0)}_{\text{PSOC}}$ is the default design recommended in Clerc and Kennedy (2002). The run length distributions shown in Figure 6.12 do not clearly indicate which configuration performs better. Although the curve of the tuned version (constriction*) is above the curve of the default variant (constriction), it is not obvious whether this difference is significant. The numerical values indicate that the tuned version performs slightly better than the default one (106.56 versus 162.02 as can be seen in Table 6.13), but the corresponding graphical representations (histograms and boxplots, not shown here) give

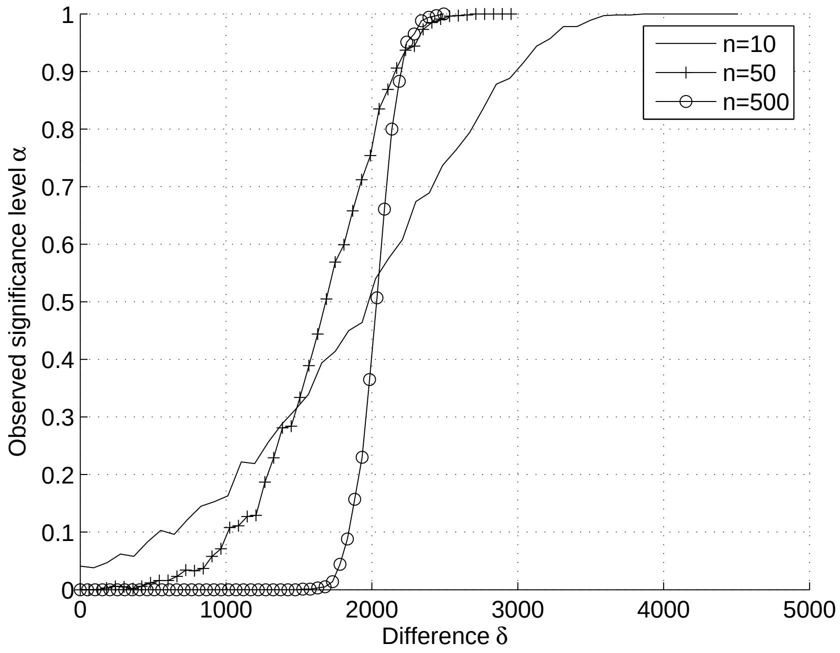


Figure 6.10: *Observed significance level. The particle swarm optimization with the default parameters is compared to the tuned version. The observed differences are* 1933*,* 1799*, and* 2077 *for* $n = 10$*,* 50*, and* 500 *experiments respectively (function value while optimizing the 10-dimensional Rosenbrock function with a budget of 2500 function evaluations). Consider* $n = 50$*: A difference as small as 1100, that would occur frequently, has an observed significance level smaller than 0.05. This is a strong indication for the assumption that there is a difference as large as 1000. This is case RE-1 as defined in Section 1.6.*

no hints that there is a significant difference between the performance of the tuned x^*_{PSOC} and $x^{(0)}_{\text{PSOC}}$ (Clerc and Kennedy, 2002). This result is not very convincing. Further investigations are recommended.

However, a t-test would accept the null hypothesis that there is no difference in means. But, is this result independent of the sample size? If the sample size is increased, for example if 2000 experiments were performed, a t-test would reject the null hypothesis at the 5 % level. This example demonstrates how the experimenter can influence the outcome of the classical t-test by varying the sample size n. Figure 6.11 illustrates the situation with tools from the arsenal of an error statistician. The result presented in Figure 6.11 is a good indication that we are observing a population where the difference in means is not larger than 120 ($n = 50$), or not larger than 50 ($n = 2000$).

But is this result really scientifically important? If the experimenter has specified the largest scientifically unimportant difference greater than zero, then this can be used to relate the statistical result to the scientific claim. Obviously, meta-statistical rules are necessary to interpret this result.

6.5.6 Comparing Particle Swarm Variants

Our next goal is to detect differences between the two major particle swarm variants, the inertia weight and the constriction factor variant. As the former requires only four parameters, a legitimate question is "Why does the inertia weight variant require three additional factors?" We consider the differences in the performance of the constriction and inertia weight particle swarm optimization variants based on optimization data from the 10-dimensional Rosenbrock function. 50 experiments were performed, resulting in 50 differences. The null hypothesis reads: "There is no difference in means." The observed difference is 66.86. As histograms and boxplots reveal, there is no statistically significant difference observable. Both configurations perform similarly. Population mean, median, and standard deviation are shown in Table 6.13. The plot of the observed significance versus the difference in means indicates that differences δ in the mean value $\tilde{f}^{(50)}$ larger than 100 occur seldom. If the experimenter specifies the smallest scientifically significant difference he can judge the consequences of accepting the null hypothesis.

6.5.7 Optimizing the Nelder-Mead Simplex Algorithm and a Quasi-Newton Method

In the Nelder-Mead simplex algorithm, four parameters must be specified, namely the coefficients of reflection, ρ, expansion, χ, contraction, γ, and shrinkage, σ. Default settings are reported in Table 5.1. Experiments indicate that the value of the reflection parameter, ρ, should be smaller than 1.5 An analysis that is based on the visualization tools from the DACE toolbox reveals that there exists a relatively small local optimum regarding χ (expansion parameter) and ψ (contraction parameter), respectively. The sequential approach could be successfully applied to the NMS algorithm, its performance on the Rosenbrock function was improved significantly. Results from this tuning process are presented in Table 6.13.

In addition to the optimization algorithms analyzed so far, the performance of a quasi-Newton method (see Section 5.2.2) was analyzed. An implementation from the commercial MATLAB optimization toolbox was used in the experiments. Quasi-Newton clearly outperformed the other algorithms, as can be seen from the results in Table 6.13.

A comparison of the RLDs of the three algorithms is shown in Figure 6.12. The results support the claim that PSO performs better than the Nelder-Mead simplex (NMS) algorithm. Only the tuned version of the latter was able to complete 20% of the experiments with success. Regarding the two PSO variants, it is not obvious which one performs better. After the tuning

Table 6.12: *PSO constriction factor. $\mathcal{D}_A$ that was used to optimize Rosenbrock's function. The variables s, χ, φ, and $v_{\max}$ have been defined in Table 5.5. $x^{(l)}_{PSOC}$ and $x^{(u)}_{PSOC}$ denote the ranges of the LHD, x^*_{PSOC} is the improved design and $x^{(0)}_{PSOC}$ is the design suggested in Clerc and Kennedy (2002).*

Design	s	χ	φ	$v_{\max}$
$x^{(l)}_{\mathrm{PSOC}}$	5	0.68	3.0	10
$x^{(u)}_{\mathrm{PSOC}}$	100	0.8	4.5	750
x^{*}_{PSOC}	17	0.759	3.205	324.438
$x^{(0)}_{\mathrm{PSOC}}$	20	0.729	4.1	100

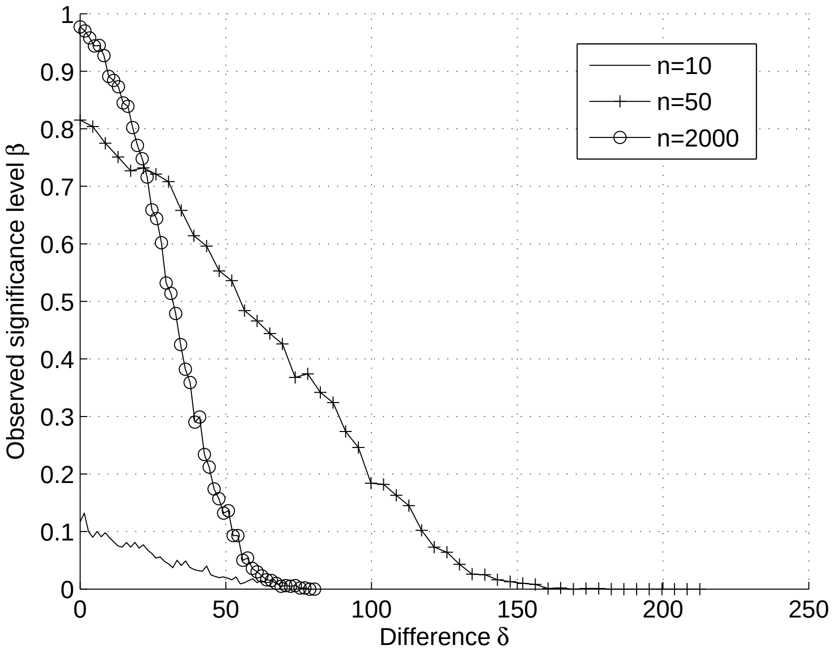


Figure 6.11: *Comparing PSO constriction and PSO constriction*, $n = 10$, 50, and 2000 repeats. The observed difference for $n = 50$ is $\overline{d} = 55.47$. A t-test would accept the null hypothesis H "there is no difference in means" for $n = 50$, because $\beta_{55.47}(0) < 0.95$. A t-test would reject the null hypothesis for $n = 2000$, because $\beta_{30.1}(0) > 0.95$). This is case AC-1 as introduced in Section 1.6. The t-test results depend on the sample size.*

process, the inertia weight variant appears to be better, but it requires the specification of seven (compared to only four in the constriction factor variant) exogenous parameters. However, the Rosenbrock function is mostly of academic interest, since it lacks many features of a real-world optimization problem.

The analysis and the tuning procedure described so far have been based solely on the average function value in 50 runs. This value may be irrelevant in a different optimization context. For example, the best function value (minimum) or the median can be alternatively used. A similar optimization procedure could have been performed for any of these cases with the presented sequential approach. Note that the optimal algorithm design presented in this study is only applicable to this specific optimization task $x^{(1)}_{\texttt{rosen}}$ as listed in Table 6.9.

As in Shi and Eberhart (1999), the starting points have been initialized randomly in the range $[15, 30]^d$. Hence, different sources of randomness are mixed in this example. The following studies will be based on deterministically generated starting points, as recommended in More et al. (1981).

6.6 Experimental Results for the S-Ring Model

The Rosenbrock function, which was considered in the previous sections, was chosen to provide a comprehensive introduction to the sequential DACE approach. In the following, we will present a more realistic real-world problem. The performance of a PSO is compared to a

Table 6.13: *Result table of the mean function values of the best particle in the swarm after $n = 50$ runs, $\tilde{f}^{(50)}$, for the Rosenbrock function. Default algorithm designs from Shi and Eberhart (1999); Clerc and Kennedy (2002); Lagarias et al. (1998), as well as the improved design for all algorithms, for $n = 50$ runs, are reported.*

Design	Mean	Median	StD	Min	Max
$x^{(0)}_{\text{PSO}}$	1.84×10^3	592.13	3.10×10^3	64.64	18519
x^{*}_{PSO}	39.70	9.44	55.38	0.79	254.19
$x^{(0)}_{\text{PSOC}}$	162.02	58.51	378.08	4.55	2.62×10^3
x^{*}_{PSOC}	106.56	37.65	165.90	0.83	647.91
$x^{(0)}_{\text{NMS}}$	9.07×10^3	1.14×10^3	2.50×10^4	153.05	154966
x^{*}_{NMS}	112.92	109.26	22.13	79.79	173.04
Quasi-Newton	5.46×10^{-11}	5.79×10^{-11}	8.62×10^{-12}	1.62×10^{-11}	6.20×10^{-11}

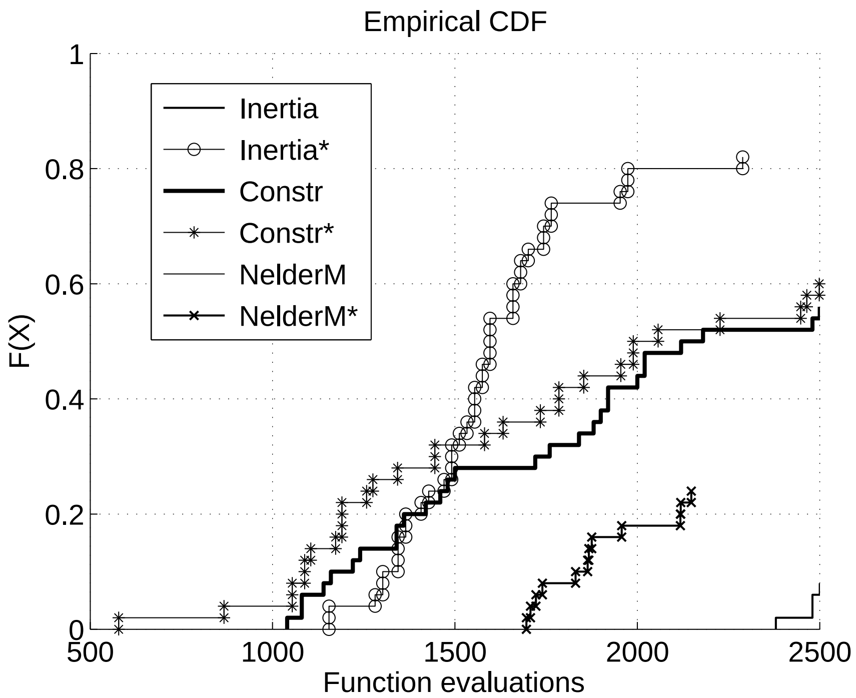


Figure 6.12: *Run length distribution. Step (S-11), the final comparison of the canonical and the improved design based on RLDs. Asterisks denote improved configurations. The improved inertia weight version of PSO succeeded in more than* 80% *of the experiments with less than* 2500 *function evaluations. The standard NMS algorithm failed completely (hence the corresponding curve is not shown in this figure), but it was able with an improved design to succeed in* 10% *of the runs after* 2500 *function evaluations. For a given budget of* 1500 *function evaluations, both the constriction factor and the improved inertia weight PSO variants perform equally well.*

NMS algorithm and to a quasi-Newton method. The S-ring simulator was selected to define a 12-dimensional optimization problem with noisy function values. The number of function evaluations, $t_{\max}$, was limited to 1000 for each optimization run. This value appears to be realistic for real-world applications. The related problem design is reported in Table 6.14.

Similar to the analysis for the Rosenbrock function, the constriction factor and inertia weight variants of PSO were analyzed. The former requires only 4 exogenous strategy parameters, whereas 7 parameters have to be specified for the latter. Optimizing the PSO inertia weight variant improved the algorithm's robustness as reported in Table 6.15. The average function value decreased from 2.61 to 2.51, which is a significant difference. However, it is very important to note that the minimum function value could not be improved, but increased slightly from 2.4083 to 2.4127. The tuning procedure was able to find an algorithm design that prevents outliers and produces robust solutions at the cost of an aggressive exploratory behavior. However, an increased probability of finding a solution that has a minimum function value could have been specified as an optimization goal, resulting in a different "optimal" design. Measures such as the best solution from n runs are better suited for real-world optimization problems than the mean function value. Computer intensive methods facilitate the determination of related statistics.

Although the function values look slightly better, the tuning process produced no significant improvement for the rest of the algorithms. The constriction factor PSO variant, as well as the NMS algorithm and the quasi-Newton method were not able to escape from local optima. In contrast to the Rosenbrock function, many real-world optimization problems have many local minima on flat plateaus. The distribution of local optima in the search space is unstructured. Therefore these algorithms were unable to escape plateaus of equal fitness. This behavior occurred independently from the parameterization of their exogenous strategy parameters. The inertia weight PSO variant that required the determination of 7 exogenous strategy parameters, outperformed the other algorithms in this comparison. Whether this improved result was caused by the scaling property of the inertia weight is subject to further investigation.

Experimental results indicate that there is no generic algorithm that works equally well on each problem. Even different instances of one problem may require different algorithms, or at least different parameterizations of the employed algorithms. None of the algorithms has proved in our study to be satisfying for every problem. The quasi-Newton method, as expected, outperformed the other algorithms on the Rosenbrock function, but it failed completely on the elevator optimization problem, where the inertia weight PSO variant, which requires nearly twice as many parameters as the PSO constriction factor variant, performed best.
Finally, we note that the determination of a good initial algorithm (and problem) design is not trivial, and therefore a drawback of the proposed approach. This is common to all statistical methods in this field, especially for the classical DOE approach.

Table 6.14: *Problem design for the S-ring experiments. Note that due to the stochastic nature of the S-ring simulation model, no additional noise was added to the function values.*

Design	n	$t_{\max}$	d	Init.	Term.	x_l	x_u	Perf.
$x^{(1)}_{\texttt{sring}}$	50	1000	12	DETMOD	EXH	-10	10	MBST

Table 6.15: *Results from the optimization of the S-ring model. Default designs, reported in Shi and Eberhart (1999); Clerc and Kennedy (2002); Lagarias et al. (1998), and improved designs, for* $n = 50$ *repeats, are reported. The tuned inertia weight PSO variant appears to be more robust than the default variant. It generates no outliers, as can be seen in the last column, but it was not able to find a better minimum.*

Design	Mean	Median	StD	Min	Max
$x^{(0)}_{\text{PSO}}$	2.6152	2.5726	0.4946	2.4083	5.9988
x^{*}_{PSO}	2.5171	2.5112	0.0754	2.4127	2.6454
$x^{(0)}_{\text{PSOC}}$	4.1743	2.6252	1.7021	2.5130	5.9999
x^{*}_{PSOC}	4.1707	2.6253	1.7055	2.5164	5.9999
$x^{(0)}_{\text{NMS}}$	4.3112	4.3126	1.7059	2.6200	5.9999
Quasi-Newton	4.3110	4.3126	1.7060	2.6186	5.9999

6.7 Criteria For Comparing Algorithms

Nowadays it is widely accepted that there is no algorithm that performs on average better than any other algorithm. Schwefel (1995) comments on evolution strategies:

> So, is the evolution strategy the long-sought-after *universal* method of optimization? Unfortunately, things are not so simple and this question cannot be answered with a clear "yes."

Some optimization algorithms are exceptionally popular, for example the Nelder-Mead simplex algorithm or evolutionary algorithms. The popularity of these direct search algorithms is not founded on their overall optimality, but might be related to the following reasons:

1. Direct search algorithms are appealing, because they are easy to explain, understand and implement. They share this feature with some of the designs presented in Chapter 4.

2. For many real-world optimization problems, it is vital to find an improvement, but not the global optimum. Direct search algorithms produce significant improvements during the first stage of their search.

3. Function evaluations are extremely costly in many real-world applications. Hence, the usage of finite-gradient approximation schemes that require at least d function evaluations in every step is prohibitive (d denotes the problem dimension).

We claim that universal optimization methods are suitable tools during the first stage of an optimization process. The experimental methodology presented in this chapter provides statistical tools to detect relevant factors. It can be advantageous to combine or even to replace the universal method with small, smart, and flexible heuristics. The experimental analysis provides means for a deepened understanding of the problem, the algorithm, and their interaction as well. Learning happens and leads to a progress in science.

6.8 Summary

The ideas presented in this chapter can be summarized as follows:

1. Tuning was introduced as an optimization process.

2. Optimization relies on very restrictive assumptions. "With the possible exception of the laboratory or casino" these assumptions are met nowhere (Klein, 2002).

3. An optimization process can be regarded as a process that enables learning. This concept is related to Mayo's extension of the classical NPT approach.

4. To start the tuning process, a performance measure has to be defined. Effectivity and efficiency can guide the choice of an adequate performance measure.

5. The classical DOE approach consists of three steps: Screening, modeling, and optimization. Each step requires different experimental designs.

6. As the assumption of a linear model for the analysis of computer programs is highly speculative, a sequential approach (SPO) that combines classical and modern statistical tools has been proposed. This sequential process can be used for tuning algorithms.

7. Results that are statistically significant are not automatically scientifically meaningful. Results from the classical Neyman-Pearson theory of testing (NPT) should be complemented with NPT* tools.

8. The optimization practitioner does not always choose the absolute best algorithm. Sometimes a robust algorithm or an algorithm that provides insight into the structure of the optimization problem is preferred.

Chapter 7

Understanding

> Life is really simple, but men insist on making it complicated.
>
> Confucius

7.1 Introduction

This chapter closes the circle on the problem begun in the discussion of the Neyman-Pearson theory in Chapter 1. It demonstrates the difference between statistical testing as an automatic rule (NPT) and as a learning tool (NPT*). Automatic rules can be implemented as computer programs that generate solutions. Learning tools provide means to interpret the relevance of these results.

This chapter consists of two parts. First a classification of methods that can be integrated into the selection process of evolutionary algorithms is presented. Threshold selection is only one approach to handle the problem of noisy function evaluations. The second part presents a case study to demonstrate how NPT* tools enable an understanding of the basic principles of threshold selection.

Understanding can be seen not only as an analytic, but also as a synthetic, bottom-up approach: Starting from the very basic parts of the algorithm, new parts are added if they improve the algorithm's performance. Interactions play an important role, since some effects may appear only as a result of correlations between two or more parts. Simple algorithms often perform excellently on realistic benchmark problems (Watson et al., 1999; Whitley et al., 2002). Therefore, it might be useful to determine the essential features of algorithms.

In this chapter we present a study devoted to the problem of selection under uncertainty. First, existing approaches and theoretical results for the design and analysis of experiments for selection and screening are presented. The related statistical procedures require assumptions that are not always met in practice, especially when applied to search algorithms.

Therefore, an experimental approach is useful. In a pre-experimental study, simple configurations are tested to find out if there is any effect at all. This study enables the experimenter to define a first experimental design and to state the scientific claim more precisely.

Bechhofer et al. (1995) give an in depth presentation of methods for statistical selection, screening, and comparisons. Rubinstein (1998) and Gigerenzer and Selten (2002) discuss models of bounded rationality. The concept of simple heuristics is presented in (Gigerenzer et al., 1999).

7.2 Selection Under Uncertainty

Noise is a common factor in most real-world optimization problems. It arises from different sources, such as measurement errors in experiments, the stochastic nature of the simulation process, or the limited amount of samples gathered from a large search space. Common means used by evolutionary algorithms to cope with noise are resampling, averaging techniques based on statistical tests, local regression methods for function value estimation, or methods to vary the population size (Stagge, 1998; Beyer, 2000; Sano and Kita, 2000; Arnold, 2001; Branke et al., 2001; Bartz-Beielstein and Markon, 2004). In this thesis we concentrate our investigations on the selection process when the function values are disturbed by additive noise.

Noise that affects the object variables is not subject of our investigations. From our point of view the following case is fundamental for the selection procedure in noisy environments:

> Reject or accept a new candidate, while the available information is uncertain. Thus, two errors may occur: An α error as the probability of accepting a worse candidate due to noise and a β error, as the error probability of rejecting a better candidate.

In the context of selection and decision making the terms "candidate" and "point" will be used synonymously. A well established context where these error probabilities are analyzed is hypothesis testing as introduced in Section 2.2.

7.2.1 A Survey of Different Selection Schemes

Depending on the prior knowledge, selection schemes can be classified according to the following criteria:

1. Threshold: subset selection – indifference zone.
2. Termination: single stage – multi stage (sequential).
3. Sample size: open procedures – closed procedures.
4. Variances: known – unknown, equal – unequal.

The goal of subset selection is the identification of a subset containing the best candidate. It is related to screening procedures. *Subset selection* is used when analyzing results, whereas the *indifference zone* (IZ) approach is used when designing experiments. The sample size r is known in subset selection approaches, it is determined prior to the experiments in the indifference zone approaches.

Single stage procedures can be distinguished from *multi stage* procedures. The terms "multi stage" and "sequential" will be used synonymously. The latter can use *elimination*: If inferior solutions are detected, they are eliminated immediately. Selection procedures are *closed*, if prior to experimentation an upper bound is placed on the number of observations to be taken from each candidate. Otherwise, they are *open*. Furthermore, it is important to know whether the variance is common or known.

Our analysis is based on the following statistical assumptions. Let $\{Y_{ij}\}$, $1 \leq i \leq r$, $1 \leq j \leq k$, denote r independent random samples of observations, taken from $k \geq 2$ candidates. The Y_{ij} can denote function values taken from candidate solutions $X_1, \ldots, X_k$ or individuals (particles) of some evolutionary algorithm. Candidate X_i has a (fitness) function value with unknown mean μ_i and common unknown variance $\sigma^2_{\epsilon,i} = \sigma^2_\epsilon$, $1 \leq i \leq k$. The *ordered means* are denoted by

$$\mu_{[1]} \leq \mu_{[2]} \leq \ldots \leq \mu_{[k]}, \tag{7.1}$$

where $\mu_{[1]}$ denotes the mean of the best candidate (minimization). Generally, normal response experiments are considered.

7.2.2 Indifference Zone Approaches – A Single Stage Procedure

In the indifference zone approach, the optimization practitioner a priory specifies a value $\delta^* > 0$ representing the smallest difference worth detecting (threshold). Errors below this threshold resulting from incorrect selection are ignored. Following Bechhofer et al. (1995) we define experimental goals (G) and associated probability requirements (P). The experimental goal is related to the scientific claim, whereas the probability requirement is related to the statistical model of experimental tests, see Figure 1.2. The first experimental goal reads:

(G-1) To select the candidate associated with the smallest mean $\mu_{[1]}$.

A *correct selection* (CS) is said to have been made if (G-1) is achieved. Let δ^*, $0 < \delta^* < \infty$, be the smallest difference worth detecting. The *probability requirement* reads

(P-1) For given constants (δ^*, P^*) with $1/k < P^* < 1$, we require

$$Pr(\text{CS}) \geq P^*, \qquad \text{whenever } \mu_{[2]} - \mu_{[1]} \geq \delta^*. \tag{7.2}$$

A configuration that satisfies the *preference zone requirement*

$$\mu_{[2]} - \mu_{[1]} \geq \delta^*, \tag{7.3}$$

is said to be in the *preference zone*, otherwise it is said to be in the *indifference zone*. Indifference zone approaches are procedures that guarantee Equation 7.2. Bechhofer et al. (1995) proposed the single stage selection procedure shown in Figure 7.1 for common known variance. The procedure shown in Figure 7.1 is location invariant, only the difference in means, and not their absolute values are important. The upper-α equicoordinate critical point $Z^{(\alpha)}_{k,\rho}$, see Equation 2.4, is determined to satisfy the probability requirement (P-1) for any true configuration of means satisfying the preference zone requirement, see Equation 7.3. Under the assumptions from Section 7.2.1 is no procedure requiring fewer observations per candidate than Procedure 7.1 if the experimenter is restricted to single stage location invariant procedures that guarantee the probability requirement (P-1) (Bechhofer et al., 1995).

7.2.3 Subset Selection

Selection of a Subset of Good Candidates

A common problem for population based direct search methods is the selection of a subset of k "good" candidates out of a set of m $(1 \leq k < m)$ under uncertainty.

Gupta (1965) proposed a single stage procedure, which is applicable when the function values of the candidates are balanced and are normal with common variance. Balanced samples are those in which the candidates have an equal number of observations.

Procedure Single stage procedure

1. For the given k and specified $(\delta^*/\sigma_\epsilon, P^*)$ determine

$$r = \left\lceil 2\left(\sigma_\epsilon Z^{(1-P^*)}_{k-1,1/2}/\delta^*\right)^2 \right\rceil . \tag{7.4}$$

2. Take a random sample of r observations Y_{ij}, $1 \leq j \leq r$, in a single stage from X_i, $1 \leq i \leq k$.

3. Calculate the k sample means $\overline{y}_i = \sum_{j=1}^r y_{ij}/r$, $1 \leq i \leq k$.

4. Select the candidate that yields the smallest sample mean $\overline{y}_{[1]}$ as the one associated with the smallest sample mean $\mu_{[1]}$.

Figure 7.1: *Indifference zone approach. Single stage procedure.*

Selection of a Random-Size Subset of Good Candidates

This selection method generates a random-size subset that contains the candidate associated with the smallest true mean $\mu_{[1]}$.

(G-2) To select a (random-size) subset that contains the candidate X_i associated with $\mu_{[1]}$.

For unknown variance σ_ϵ^2 the probability of a correct selection depends on (μ, σ_ϵ^2). If the variance is known, Pr(CS) depends only on the true means $\mu = (\mu_1, \ldots, \mu_k)$.

(P-2) For a specified constant P^* with $1/k < P^* < 1$, we require that

$$Pr\{\text{CS}|(\mu, \sigma_\epsilon^2)\} \geq P^* \tag{7.5}$$

for all μ.

Bartz-Beielstein and Markon (2004) describe an implementation of this selection scheme for evolutionary algorithms in noisy environments. As the size of the selected subset is not known in advance, the population size varies during the optimization: It increases with the noise level. Nelson et al. (1998) and Goldsman and Nelson (1998) propose an extension of Gupta's single stage procedure, that is also applicable if the variances are unknown and not necessarily equal. A subset-selection approach for the selection of the k best candidates is described in Bechhofer et al. (1995, p. 86). The procedure shown in Figure 7.2 implements a (random-size) subset-selection method.

Selection of δ^*-Near-Best Candidates

Selecting the near-best candidate may be more useful than selecting the k best candidates in some situations (Fabian, 1962; Bechhofer et al., 1995). Candidate X_i is δ^*-near-best, if μ_i is within a specified amount $\delta^* > 0$ of the smallest sample mean:

$$\mu_i \leq \mu_{[1]} + \delta^*, \tag{7.11}$$

Procedure Gupta Selection

1. Take a random sample of r observations Y_{ij}, $1 \leq j \leq r$, in a single stage from X_i, $1 \leq i \leq k$.

2. Calculate the k sample means $\overline{y}_i = \sum_{j=1}^{r} y_{ij}/r$, $1 \leq i \leq k$.

3. If the variance σ_ϵ^2 is unknown, calculate

$$s_\nu^2 = \sum_{i=1}^{k} \sum_{j=1}^{r} (y_{ij} - \overline{y}_i)^2/\nu, \tag{7.6}$$

   the unbiased pooled estimate of σ_ϵ^2 based on $\nu = k(r-1)$ degrees of freedom.

4. We have to distinguish between known and unknown variance:

   (a) In case of known variance σ_ϵ^2, include the candidate X_i in the selected subset if and only if

$$\overline{y}_i \leq \overline{y}_{[1]} + h\sigma_\epsilon\sqrt{2/r}, \tag{7.7}$$

   where

$$h = Z_{k-1,1/2}^{(1-P^*)}. \tag{7.8}$$

   (b) In case of unknown variance σ_ϵ^2, include the candidate X_i in the selected subset if and only if

$$\overline{y}_i \leq \overline{y}_{[1]} + hs_\nu\sqrt{2/r}, \tag{7.9}$$

   where

$$h = T_{k-1,\nu,1/2}^{(1-P^*)}. \tag{7.10}$$

Figure 7.2: *Subset selection. Gupta selection is a single stage procedure. If σ_ϵ^2 is known, h is the upper-$(1-P^*)$ equicoordinate critical point of the the equicorrelated multivariate standard normal distribution, see Equation 2.4. If σ_ϵ^2 is unknown, h is the equicoordinate critical point of the equicorrelated multivariate t-distribution.*

(G-3) Select a (random-size) subset that contains at least one candidate X_i satisfying Equation 7.11.

(P-3) For specified constants (δ^*, P^*) with $\delta^* > 0$ and $1/k < P^* < 1$, we require that (see Roth (1978); van der Laan (1992))

$$Pr\{\delta^*\text{-near-best CS}\} \geq P^* \tag{7.12}$$

for all μ.

A δ^*-near-best selection procedure is shown in Figure 7.3.

7.2.4 Threshold Selection

Threshold rejection (TR) and *threshold acceptance* (TA) are complementary strategies. Threshold rejection is a selection method for evolutionary algorithms, that accepts new candidates if their noisy function values are significantly better than the value of the other candidates (Markon et al., 2001). "Significant" is equivalent to "by at least a margin of τ". Threshold acceptance accepts a new candidate even if its noisy function value is worse. The term *threshold selection* (TS) subsumes both selection strategies.
The basic idea of threshold selection is relatively simple and already known in other contexts:

- Matyáš (1965) introduced a threshold operator (with some errors, see Driml and Hanš (1967)) for a (1+1)-evolution strategy and objective functions without noise.
- Stewart et al. (1967) proposed a threshold strategy that accepts only random changes that result in a specified minimum improvement in the function value.
- Dueck and Scheuer (1990) presented a threshold acceptance algorithm.
- Nagylaki (1992) stated that a similar principle, the so-called truncation selection, is very important in plant and animal breeding: "Only individuals with phenotypic value at least as great as some number c are permitted to reproduce." Truncation selection is important for breeders, but it is unlikely to occur in natural populations.
- Jansen and Wegener (2000) compared the (1+1)-EA to a variant, the so called $(1+1)^*$-EA, that accepts only the offspring whose function value is strictly better (smaller) than the function value of its parent. They considered a discrete search space $\mathbb{B}^d$ and objective functions without noise.

Threshold selection is also related to Fredkin's paradox: "The more equally attractive two alternatives seem, the harder it can be to choose between them—no matter that, to the same degree, the choice can only matter less"(Minsky, 1985). Regarding the distinction between rules of inductive behavior and learning rules given in Section 1.6.2, TS as presented here is an automatic test rule and belongs to the former type of rules.

Procedure δ^*-Near-Best Selection

1. Take a random sample of r observations Y_{ij}, $1 \leq j \leq r$, in a single stage from X_i, $1 \leq i \leq k$.
2. Calculate the k sample means $\overline{y}_i = \sum_{j=1}^{r} y_{ij}/r$, $1 \leq i \leq k$.
3. Include the candidate X_i in the selected subset if and only if

$$\overline{y}_i \leq \overline{y}_{[1]} + h(\delta^*)\sigma_\epsilon \sqrt{2/r}, \tag{7.13}$$

where

$$h(\delta^*) = Z_{k-1,1/2}^{(1-P^*)} - \delta^*/\sigma_\epsilon \sqrt{r/2}. \tag{7.14}$$

Figure 7.3: *δ^*-Near-Best Selection*

Procedure Threshold Selection

1. Given: A candidate X_1 with a related sample Y_{1j} of r observations and sample mean $\overline{y}_1 = \sum_{j=1}^{r} y_{1j}/r$.
2. Take a random sample of r observations Y_{2j}, $1 \leq j \leq r$, in a single stage from a new candidate X_2.
3. Calculate the sample mean $\overline{y}_2 = \sum_{j=1}^{r} y_{2j}/r$.
4. Select the new candidate X_2 if and only if

$$TR : \overline{y}_2 + \tau < \overline{y}_1, \qquad \text{with } \tau \geq 0 \tag{7.15}$$

or

$$TA : \overline{y}_2 + \tau < \overline{y}_1, \qquad \text{with } \tau \leq 0. \tag{7.16}$$

Figure 7.4: *Threshold selection. This basic procedure can be implemented in many optimization algorithms, for example evolution strategies or particle swarm optimization.*

The Threshold Selection Procedure

As in (G-1), the experimental goal is to select the candidate associated with the smallest mean $\mu_{[1]}$. Figure 7.4 shows the threshold selection algorithm. As can be seen from Equation 7.15, threshold rejection increases the chance of rejecting a worse candidate at the expense of accepting a good candidate. It might be adequate if there is a very small probability of generating a good candidate. Equation 7.16 reveals that threshold acceptance increases the chance of accepting a good candidate at the risk of failing to reject worse candidates.

Threshold Selection and Hypothesis Testing

The calculation of a threshold value for the TR scheme can be interpreted in the context of hypothesis testing as the determination of a critical point (Beielstein and Markon, 2001). The critical point $c_{1-\alpha}$ for a hypothesis test is a threshold to which one compares the value of the test statistic in a sample. It specifies the critical region CR and can be used to determine whether or not the null hypothesis is rejected. We are seeking a value $c_{1-\alpha}$, so that

$$Pr\{S > c_{1-\alpha} \quad |H \text{ true } \} \leq \alpha, \tag{7.17}$$

where S denotes the test statistic and the null hypothesis H reads: "There is no difference in means." The threshold acceptance selection method can be interpreted in a similar manner.

Generally, hypothesis testing interpreted as an automatic rule as introduced in Section 1.6.2 considers two-decision problems in which a null hypothesis is either accepted or rejected. A false null hypothesis H can be rejected 50 % of the time by simply tossing a coin. Every time that heads comes up, H is rejected. The rejection procedures considered so far (Procedures 7.1-7.3) can be applied to k-decision problems. Here, larger sample sizes are required than for the two-decision problem. The probability of a correct selection for $k > 2$ is smaller than 50 % if the decision is based on the roll of a fair k-sided die. To avoid too large sample sizes r for fixed k the indifference zone δ^* can be increased or the probability of a correct selection P^* can be reduced.

Known Theoretical Results

The theoretical analysis in Markon et al. (2001), where threshold rejection was introduced for evolutionary algorithms with noisy function values, was based on the progress rate theory on the sphere model and was shown for the $(1+1)$-evolution strategy. However, this theoretical result is only applicable when the distance to the optimum and the noise level are known—conditions that are not very often met in practice. By interpreting this result qualitatively, we can see that the threshold value τ should be increased while approaching the optimizer x^* (τ should be infinite, when the optimum is obtained).

Another approach was used by Beielstein and Markon (2001). They demonstrated theoretically and experimentally how threshold rejection can improve the quality gain introduced in Equation 6.4 as the expected change in the function value). The influence of TR on the selection process was analyzed using a simple stochastic search model that is related to a model proposed in Goldberg (1989). This model possesses many crucial features of real-world optimization problems, i. e. a small probability of generating a better offspring in an uncertain environment. Then the search can be misled, although the algorithm selects only "better" candidates. TR can prevent this effect. In the simple stochastic search model the optimal

threshold value could be calculated as a function of the noise strength, the probability of generating a better candidate, and the difference between the expectation of the function values of two adjacent states.
However, the determination of an optimal threshold value in this simple search model required information that is usually not available in real-word situations. For example, the probability of generating a better offspring is unknown during the search process.

7.2.5 Sequential Selection

The selection procedures presented above can be extended to sequential strategies. We list some promising approaches that might be applicable as population based selection schemes.

For the case of unknown variance σ_ϵ^2, Santner (1976) proposed a two-stage selection scheme. Considering candidates with different means, unknown and not necessarily equal variances, Sullivan and Wilson (1984, 1989) presented a bounded subset selection for selecting a δ^*-near-best candidate. A fully sequential procedure was proposed by Paulson (1964). Hartmann (1988, 1991) improved this procedure. Kim and Nelson (2001) extended the approach from Hartmann (1991) to unequal and unknown variances.

Bechhofer et al. (1990) and Kim and Nelson (2001) demonstrate the superiority of sequential selection methods over two-stage-ranking-and-selection procedures. Pichitlamken and Nelson (2001) and Pichitlamken et al. (2003) present a sequential selection procedure with memory (SSM). SSM is fully sequential with elimination: If inferior solutions are detected, they are eliminated immediately.

Population based algorithms can benefit from Welch's approximation. The original goal of Welch's approximation was to solve the Behren-Fisher problem (Welch, 1947). A Student's t distribution was used to approximate the distribution of the standardized difference between sample means. Welch's approximation avoids storing the function values from previous generations.

7.3 A Case Study: Threshold Selection Applied to Evolution Strategies

This section describes how selection methods introduced in Section 7.2 can be integrated into optimization algorithms. We will present an experimental approach to analyze the effect of these methods that is not limited to situations where the distance to the true optimum or the probability of generating a better offspring as in Markon et al. (2001) or Beielstein and Markon (2001), respectively, is known.

7.3.1 Pre-Experimental Studies

Beielstein et al. (2003a) only demonstrated the effect of threshold selection in a complex optimization environment. Our goal is to understand the effect of threshold selection on the algorithm's performance.

To analyze the influence of threshold rejection we use a bottom-up approach. Therefore, an elementary optimization algorithm, the $(1 + 1)$-evolution strategy, and a simple test function, the `sphere` function, are chosen for the first experiments. To avoid floor or ceiling effects, the run-length distribution of an (1+1)-ES on the `sphere` function without noise has been studied.

Table 7.1: *Problem design for the pre-experimental* $(1+1)$*-ES runs. This table shows the experiment number, the number of repeat runs* n*, the maximum number of function evaluations* $t_{\max}$*, the problem dimension* d*, the initialization method as defined in Section 4.6.1, the termination criterion, lower* x_l *and upper* x_u *bounds for the initialization, the optimization problem as defined in Table 3.1, the performance measure from Section 6.3, and the noise level* σ_ϵ^2*.*

Design	n	$t_{\max}$	d	Init.	Term.	x_l	x_u	Perf.	Noise
$x^{(0)}_{\texttt{sphere}}$	5	1000	2	DETMOD	EXH	15	30	MBST	$0-50$

The corresponding problem design ($x^{(1)}_{\texttt{sphere}}$) is shown in Table 7.1. The first experiments have been performed to analyze the hypothesis

(H-7.1) Threshold selection affects the performance of the $(1+1)$-ES.

The procedure shown in Figure 7.5 is an implementation of the $(1+1)$-evolution strategy with threshold selection. The impact of the selection scheme on the performance was measured for different noise levels. Results from this pre-experimental study indicate that threshold selection can improve the search. But, we did only observe this effect—it could not be explained yet. And we did not prove that the threshold selection procedure caused this effect. This problem is related to the ceteris paribus conditions mentioned in Chapter 2. Consequently we have to analyze further aspects of the selection process.

7.3.2 A Simulation Study

Instead of analyzing the global performance, i.e. the mean function values after 1000 function evaluations, we start with an analysis of the local performance. Simulations were performed to analyze the influence of the threshold value on the progress rate φ (PRATE) and on the success rate s_r as defined in Section 6.3. Figure 7.6 describes the simulation procedure. The problem designs are shown in Table 7.2. The corresponding hypothesis reads:

(H-7.2) Threshold selection produces better quality results than plus selection, if the function values are disturbed by additive, gaussian noise. The results are independent of the test problem.

To reject hypothesis (H-7.2), we have to find a test function on which the $(1+1)$-ES performs better than the $(1+1)$-TS. We consider three candidates: The absolute value function (`abs`), the identity function (`id`), and the sphere function (`sphere`). A constant noise level $\sigma_\epsilon^2 = 1$,

Table 7.2: *Problem designs for the* $(1+1)$*-ES simulation runs. The progress rate* PRATE *was chosen as a performance measure. Note, that the sample size is denoted by* r*. The starting point is chosen deterministically:* $x^{(0)} = 1$*.*

Design	r	$t_{\max}$	d	Init.	Term.	$x^{(0)}$	Perf.	Noise
$x^{(1)}_{\texttt{abs}}$	10^5	1	1	DETEQ	EXH	1	PRATE	1
$x^{(1)}_{\texttt{id}}$	10^5	1	1	DETEQ	EXH	1	PRATE	1
$x^{(1)}_{\texttt{sphere}}$	10^5	1	1	DETEQ	EXH	1	PRATE	1

Procedure $(1 + 1)$-ES with threshold selection.

Initialization: Initialize the generation counter: $g = 0$. Determine a threshold value $\tau^{(g)}$. Determine a point $X_1^{(g)}$ and a standard deviation $\sigma^{(g)}$ with associated position vector $x_1^{(g)} \in \mathbb{R}^d$. Determine the (noisy) function value $\tilde{y}_1 = \tilde{f}(x_1^{(g)})$.

`repeat`

Mutation: Generate a new point $X_2^{(g)}$ with associated position vector $x_2^{(g)}$ as follows:

$$x_2^{(g)} = x_1^{(g)} + z, \tag{7.18}$$

where z is a d-dimensional vector. Each component of z is the realization of a normal random variable Z with mean zero and standard deviation $\sigma^{(g)}$.

Evaluation: Determine the (noisy) function value $\tilde{y}_2 = \tilde{f}(x_2^{(g)})$.

Selection: Accept $X_2^{(g)}$ as $X_1^{(g+1)}$ if the sum of its noisy function value and the threshold value $\tau^{(g)}$ does not exceed that of $X_1^{(g)}$:

$$\tilde{y}_2 + \tau^{(g)} < \tilde{y}_1, \tag{7.19}$$

otherwise retain $X_1^{(g)}$ as $X_1^{(g+1)}$.

Adaptation: Update $\sigma^{(g)}$ and $\tau^{(g)}$. Increment g.

`until` some stopping criterion is satisfied.

Figure 7.5: *The* $(1+1)$*-ES with threshold selection. This is an extension of the* $(1+1)$*-ES presented in Figure 5.2.*

Procedure $(1+1)$-ES simulation to approximate the one-generation progress φ.

Initialization: Initialize the sample counter $i = 1$. The index i has been suppressed to improve readability. Choose one initial parent X_1 with associated position vector $x_1 \in \mathbb{R}^d$. Choose the standard deviation $\sigma \in \mathbb{R}_+$, the threshold value $\tau \in \mathbb{R}$, and the noise level σ_ϵ^2.

`repeat`

Mutation: Generate a new point X_2 with position vector x_2 as follows:

$$x_2 = x_1 + z, \tag{7.20}$$

where each component of the d-dimensional vector z is the realization of random variable $Z \sim \mathcal{N}(0, \sigma^2)$.

Evaluation: Determine the function values

$$y_j = f(x_j), \tag{7.21}$$

$$\tilde{y}_j = f(x_j) + w_j, \tag{7.22}$$

where w_j are realizations of $\mathcal{N}(0, \sigma_\epsilon^2)$ distributed random variables W_j, $j = 1, 2$.

Selection: Accept X_2 if

$$\tilde{y}_2 + \tau < \tilde{y}_1, \tag{7.23}$$

otherwise reject X_2.

Progress: Determine

$$\delta_i = \begin{cases} x_1 - x_2, & \text{if } X_2 \text{ was accepted,} \\ 0, & \text{otherwise.} \end{cases} \tag{7.24}$$

Increment i.

`until` r samples have been obtained.

Return $\sum_{i=1}^r \delta_i / r$, an estimate of the expected progress φ from generation g to $g+1$, see (PM-14) in Section 6.3.

Figure 7.6: *$(1+1)$-ES simulation to study the effect of threshold selection on the progress rate φ.*

the starting point $x^{(0)} = 1$, and the step size $\sigma = 1$ have been chosen for these experiments. Figure 7.7 illustrates the results. The approximated progress rate φ is plotted against the threshold value τ. Positive φ values are better, because φ is the expected change in the distance of the search point to the optimum in one generation. The results from this study show that threshold acceptance ($\tau \leq 0$) can improve the progress rate on the absolute value function and on the sphere function. But threshold acceptance worsens the performance on the identity function (`id`). And threshold rejection ($\tau > 0$) worsens the progress rate in any case.

What are the differences between `id`, `abs`, and `sphere`? The starting point $x^{(0)}$ was chosen in the immediate vicinity of the global minimizer x^* of the test functions `abs` and `sphere`. This closeness to the optimum might explain the effect of the threshold selection scheme. This consideration leads to the next hypothesis:

(H-7.3) Threshold selection produces better quality results than plus selection in the vicinity of the global minimizer x^*, if the function values are disturbed by additive, gaussian noise. The results are independent of the test problem.

The problem design in Table 7.3 was used to perform the experiments. As before, the simulation procedure shown in Figure 7.6 was used to approximate the one-step progress rate φ. The results (not shown here) indicate that the starting point $x^{(0)}$ influences the threshold selection scheme. The optimal threshold value decreases (becomes negative) as the distance of the starting point $x^{(0)}$ to the optimum x^* is reduced.

A similar effect could be observed for the absolute value function `abs`. The influence of the TS scheme vanishes already if the starting point $x^{(0)} = 2$ was chosen.

Both functions, `abs` and `sphere`, are convex. Recall that a convex function is a continuous function whose value at the midpoint of every interval in its domain does not exceed the average of its values at the ends of the interval (Weisstein, 2004). In other words, a function $f(x)$ is convex on an interval $[a, b]$ if for any two points x_1 and x_2 in $[a, b]$,

$$f\left(\frac{1}{2}(x_1 + x_2)\right) \leq \frac{1}{2} f(x_1 + x_2).$$

A function $f(x)$ is strictly convex if $f\left(\frac{1}{2}(x_1 + x_2)\right) < \frac{1}{2} f(x_1 + x_2)$. I.e., a function is convex if and only if its epigraph (the set of points lying on or above the graph) is a convex set. The

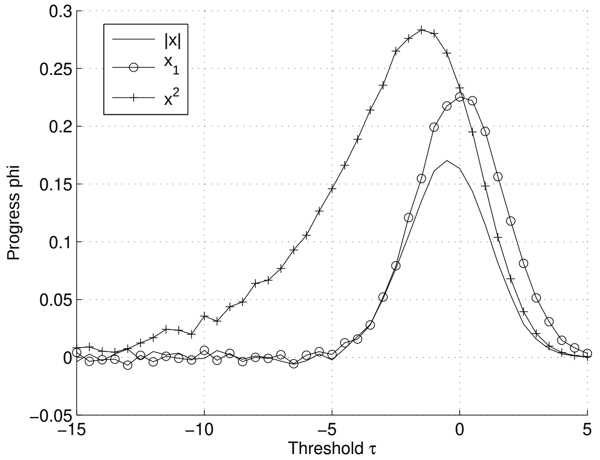


Figure 7.7: *Three different functions:* `abs`, `id`, *and* `sphere` *and related problem designs* $x^{(1)}_{\texttt{abs}}$, $x^{(1)}_{\texttt{id}}$, *and* $x^{(1)}_{\texttt{sphere}}$, *respectively, from Table 7.2. Results from the simulation study described in Figure 7.6 to study the effect of* TS *on the progress rate. Progress rate* φ *plotted against threshold value* τ*. Noise* $\sigma^2_\epsilon = 1$*, starting point* $x^{(0)} = 1$*, and step size* $\sigma = 1$*. These results indicate that threshold selection produces worse results on the identity function, whereas positive effects could be observed on* $x^{(1)}_{\texttt{abs}}$ *and* $x^{(1)}_{\texttt{sphere}}$*.*

Table 7.3: *Problem design for the (1 + 1)-ES simulation runs. The distance to the optimum of the starting point is varied.*

Design	r	$t_{\max}$	d	Init.	Term.	x_l	x_u	Perf.	Noise
$x^{(2)}_{\texttt{abs}}$	10^5	1	1	DETMOD	EXH	1	10	PRATE	1
$x^{(2)}_{\texttt{id}}$	10^5	1	1	DETMOD	EXH	1	10	PRATE	1
$x^{(2)}_{\texttt{sphere}}$	10^5	1	1	DETMOD	EXH	1	10	PRATE	1

sphere function x^2 and the absolute value function $|x|$ are convex. The function $\texttt{id}(x) = x$ is convex but not strictly convex. Thus, the next hypothesis reads:

(H-7.4) Let f denote a strictly convex test function. Threshold acceptance produces better quality results than plus selection in the vicinity of the global minimizer x^* of f if the function values are disturbed by additive, gaussian noise.

To test hypothesis (H-7.4), we simulate the (1 + 1)-ES on the bisecting line cosine function (`bilcos`). This function has infinitely many local minimizers $x_i = 2i + \epsilon$ and infinitely many local maximizers $x_i = 2i - 1 - \epsilon$, with $i \in \mathbb{Z}$ and $\epsilon = \sin^{-1}(1/\pi)/\pi \approx -.1031$.

Figure 7.8 illustrates the results from these simulations: The (1 + 1)-ES with threshold acceptance performs better with threshold acceptance if a starting point $x^{(0)}$ in the neighborhood of a local minimum is chosen. Threshold rejection improves the approximated progress rate in the neighborhood of a local maximum. A zero threshold value is best if $x^{(0)}$ is placed between two local optima. This simulation study demonstrated that the curvature influences the optimal threshold value: The (1 + 1)-ES with threshold acceptance performs better on strictly convex functions than the (1+1)-ES, whereas the (1+1)-ES with threshold rejection performs better than the (1 + 1)-ES on strictly concave functions. The next experiments to refine our hypothesis are conducted to analyze the influence of the noise level σ^2_ϵ. Therefore, we state:

(H-7.5) Let f (g) denote a strictly convex (concave) test function. Threshold acceptance (rejection) produces better quality results than plus selection in the vicinity of a local minimum (maximum) of f (g) if the function values are disturbed by additive, gaussian noise. The optimal threshold value τ^* increases as the noise level σ^2_ϵ grows.

Experiments varying the noise level σ^2_ϵ (not presented here) gave no indication that (H-7.5) is wrong. If the noise level is very high, threshold selection cannot improve the progress rate. The influence of the TS is only marginal for small noise levels.

7.3.3 Plateaus and Discrete Test Functions

To analyze the influence of threshold selection on the algorithm's performance for functions with large plateaus as in the S-ring or the elevator control optimization, we can use floor or ceiling functions. Consider the step function $\texttt{step}(x) = \lfloor x \rfloor$. The center point of the i-th plateau is defined as $x_{c,i} := i - 1/2$. If the center point of a plateau is chosen as the starting point $x^{(0)}$, then threshold selection worsens the progress rate. If $x^{(0)} \in [i - 1, x_{c,i}]$, then exists a positive threshold value that improves the progress rate (TR). If $x^{(0)} \in [x_{c,i}, i]$, then exists a negative threshold value that improves the progress rate (TA). This result is in

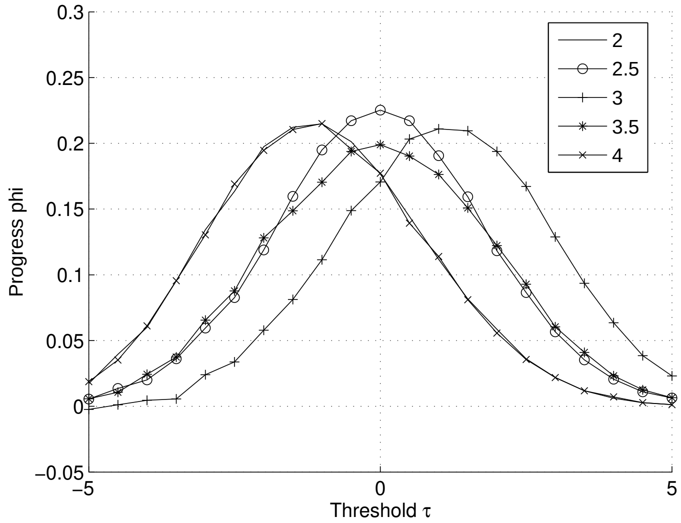


Figure 7.8: *Progress rate φ and threshold selection. Bisecting line cosine function* `bilcos`*. Curves represent results from experiments with different starting points $x^{(0)} \in \{2, 2.5, 3, 3.5, 4\}$. A positive threshold value improves the performance if the simulation is started from a local maximum, i.e. $x^{(0)} = 3$.*

correspondence with the result from Beielstein and Markon (2001), where the effect of TS on a discrete function $f(i) = i$, $i \in \mathbb{Z}$, was analyzed. In this case threshold rejection improved the progress rate. Results from the studies presented in Beielstein and Markon (2001) could be reproduced exactly in the $(1 + 1)$-ES simulation model from Figure 7.6.

The experimental analysis of the simulation model produced important results that enable an understanding of the influence of threshold selection on the local performance of the $(1 + 1)$-ES. Before we investigate the global performance, we present a theoretical analysis that complements the experimental results.

7.3.4 An Exact Analysis of the Local Performance

We discuss some results from the identity function $\mathtt{id}(x) = x$ and from the absolute value function $\mathtt{abs}(x) = |x|$. The essential behavior of optimization algorithms can be analyzed exactly only on simple test functions with the methodology presented here, because the required integrals become increasingly more difficult.

The Identity Function

Regarding the identity function, the expected one-generation progress φ of the $(1 + 1)$-ES with threshold selection (Figure 7.5) can be determined as

$$\varphi = \frac{1}{4\pi\sigma\sigma_\epsilon} \int_{z=-\infty}^{\infty} \int_{w=z+\tau}^{\infty} -z f_1(w) f_2(z)\, dw\, dz, \tag{7.25}$$

with

$$f_1(w) = \exp\left\{ -\frac{1}{2} \left(\frac{w}{2\sigma_\epsilon} \right)^2 \right\} \tag{7.26}$$

and

$$f_2(z) = \exp\left\{ -\frac{1}{2} \left(\frac{z}{\sigma} \right)^2 \right\}. \tag{7.27}$$

A perspective plot of the joint density of W and Z is shown for $\sigma = 1$ and $\sigma_\epsilon^2 = 2$ in Figure 7.9. Note, that $-z$ is used to determine the expected progress $E(\varphi_x)$ in Equation 7.25, because we are considering minimization problems. Equation 7.25 is easy to understand. Consider the $(1+1)$-ES simulation procedure described in Figure 7.6. According to Equation 7.23 the probability that the new candidate is accepted reads

$$\begin{aligned} Pr(\tilde{y}_2 + \tau < \tilde{y}_1) &= Pr(Z < W_1 + W_2 - \tau) \\ &= \frac{1}{\sqrt{2\pi}\sigma 2\sqrt{2\pi}\sigma_\epsilon} \int_{z=-\infty}^{\infty} \int_{w=z+\tau}^{\infty} f_1(w) f_2(z)\, dw\, dz. \end{aligned}$$

Analyzing this distribution is the key to the understanding of several important properties of the threshold selection process. Denote the size of the region A by $F(A)$. Consider Figure 7.10. The line g_1 divides the plane J that is spanned by the random variables w and z into the upper half plane J^+ and the lower half plane J^-. The product of the volume under the density surface over J^- and z is the expected progress $E(\varphi_x)$. The region of interest J^- is divided by the $w-$axis into the regions K^+ and K^- with negative and positive expected progress respectively:

$$\iint\limits_{K^+} f_1(w) f_2(w)\, dw\, dz \leq 0, \text{ and } \iint\limits_{K^-} f_1(w) f_2(w)\, dw\, dz \geq 0.$$

A geometric argument shows that

$$F(I_0^*) = F(I_0^+), \tag{7.28}$$

and the product of the volume under the density surface over I_0^- and $-z$ equals $E(\varphi_x)$. Introducing threshold rejection ($\tau > 0$) moves the point S along the w-axis to the right, threshold acceptance moves S to the left. A configuration for $\tau < 0$ is illustrated in Figure 7.10. One can see that $\tau = 0$ maximizes the expected progress:

$$F(I_0^-) \geq F(I_\tau^-) \quad \forall \tau \in \mathbb{R}. \tag{7.29}$$

Threshold selection worsens the expected progress on the identity function `id` under additive, gaussian noise. This result is consistent with the conclusions drawn from the experimental study.

The Absolute Value Function

The determination of the expected one-generation progress $\varphi_{|x|}$ for the absolute value function is slightly more complicated. The expected one-generation progress reads

$$\varphi = \frac{1}{4\pi\sigma\sigma_\epsilon} \int_{w=s}^{\infty} \int_{z=b}^{a} -z f_1(w) f_2(z)\, dz\, dw, \tag{7.30}$$

with $s = \tau - x^{(0)}$, $a = w - \tau$, and $b = -w + \tau - 2x^{(0)}$. Equation 7.30 can be derived as follows: According to Equation 7.23 the probability that the new candidate is accepted reads for the absolute value function `abs`

$$\begin{aligned} Pr(\tilde{y}_2 + \tau < \tilde{y}_1) &= Pr(|x^{(0)} + Z| + W_2 + \tau < |x^{(0)}| + W_1) \qquad (7.31) \\ &= \frac{1}{\sqrt{2\pi}\sigma 2\sqrt{2\pi}\sigma_\epsilon} \int_{w=s}^{\infty} \int_{z=b}^{a} \exp\left\{-\frac{1}{2}\left(\frac{w}{2\sigma_\epsilon}\right)^2\right\} \exp\left\{-\frac{1}{2}\left(\frac{z}{\sigma}\right)^2\right\} dz\, dw. \end{aligned}$$

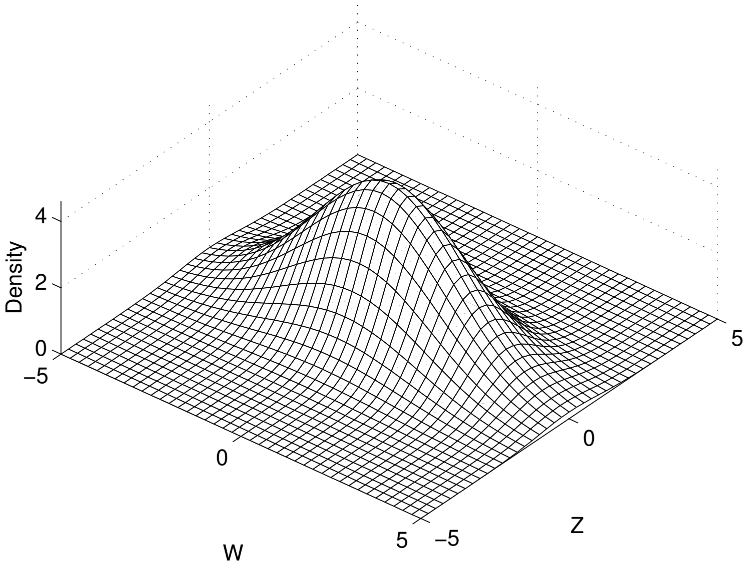


Figure 7.9: *Perspective plot of the joint density of W and Z. The density is given by $f(w,z) = f_1(w)f_2(z)$, with f_1 and f_2 as defined in Equations 7.26 and 7.27.*

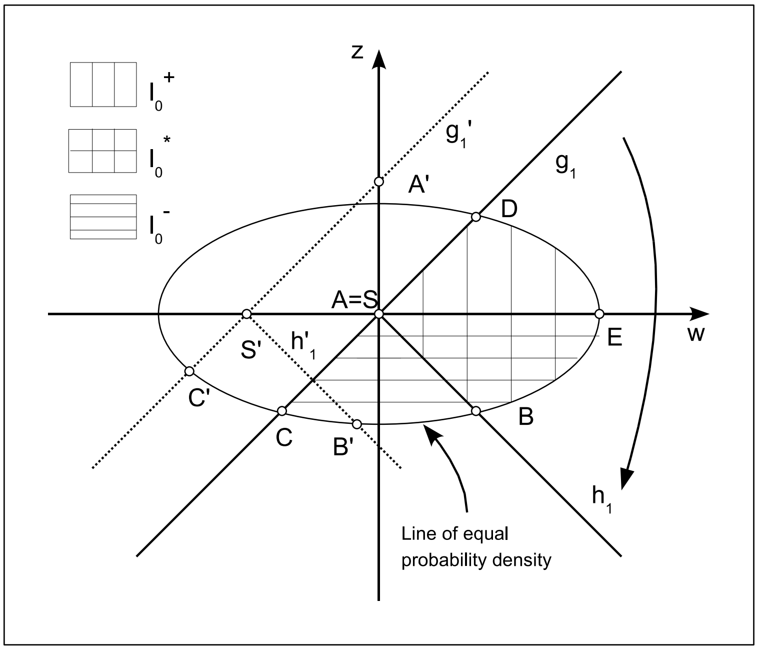


Figure 7.10: *The influence of threshold selection on the expected progress $E(\varphi_x)$ of an $(1+1)$-ES on the identity function. A threshold value $\tau \neq 0$ can only worsen the expected progress $E(\varphi_x)$, no matter what values are chosen for the step size σ or noise level σ_ϵ^2.*

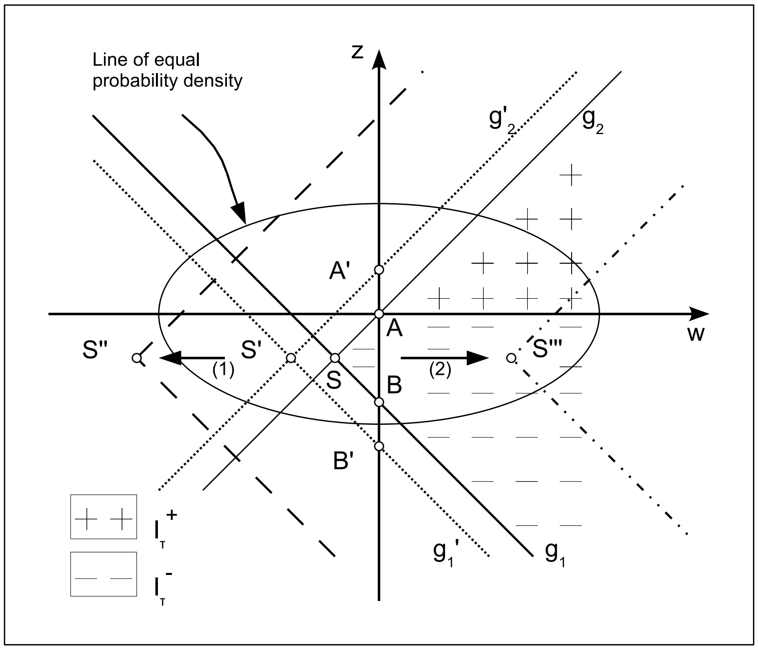


Figure 7.11: *The influence of threshold selection on the expected progress $E(\varphi_{|x|})$ of an $(1+1)$-ES on the absolute value function. For large $x^{(0)}$ values, the situation is similar to the situation illustrated in Figure 7.10. The coordinates read: $A = (0,0)$, $A' = (0,-\tau)$, $B = (0,-2x^{(0)})$, and $B' = (0,\tau - 2x^{(0)})$.*

The expectation is calculated by integration over the wedge shaped area I_τ shown in Figure 7.11. The relevant area I_τ is splitted by the w-axis into a positive and a negative part: $I_\tau = I_\tau^+ + I_\tau^-$. Hence:

$$\iint\limits_{I_\tau} -zf_1(w)f_2(z)\,dz\,dw = \iint\limits_{I_\tau^+} -zf_1(w)f_2(z)\,dz\,dw + \iint\limits_{I_\tau^-} -zf_1(w)f_2(z)\,dz\,dw$$
$$= \int_s^\infty \int_b^0 -zf_1(w)f_2(z)\,dz\,dw + \int_s^\infty \int_0^a -zf_1(w)f_2(z)\,dz\,dw.$$

Consider the points $A = (0,0)$, $A' = (0,-\tau)$, $B = (0,-2x^{(0)})$, $B' = (0,\tau - 2x^{(0)})$, S, and S', see Figure 7.11. Threshold acceptance ($\tau < 0$) moves A to A', B to B', and S to S'. Therefore the wedge shaped area I_τ is moved to the left as denoted by the arrow (1). As I_τ^+ and I_τ^- are non-symmetric with respect to the w-axis, the expected progress $E(\varphi_{|x|})$ can be improved for certain τ values. However,

$$\iint\limits_{I_\tau^+} -zf_1(w)f_2(z)\,dz\,dw = \iint\limits_{I_\tau^-} -zf_1(w)f_2(z)\,dz\,dw$$

as $\tau \to \pm\infty$, and therefore $E(\varphi_{|x|})$ approaches zero for too large τ-values.

We can see even more from Figure 7.11. Note, that $E(\varphi_{|x|})$ depends on $x^{(0)}$. If $x^{(0)}$ is increased, the influence of the straight line g_1 on the progress vanishes. Hence, the absolute value function behaves as the identity function $f(x) = x$ for large $x^{(0)}$ values (far away from the optimum x^*).

Results from the theoretical analysis of the dependency of $E(\varphi_{|x|})$ on $x^{(0)}$ have been confirmed experimentally. Furthermore, the optimal threshold value can be determined numerically. For example, if the noise level reads $\sigma_\epsilon^2 = 1$ and the starting point $x^{(0)} = 1$, the optimal threshold value is $\tau^* = -0.42$. The optimal threshold value approaches zero as the distance of the starting point to the optimum is increased. These results are in correspondence with hypothesis (H-7.5). Equation 7.31 might enable a further theoretical analysis of the relationship of convexity and threshold selection.

7.3.5 Summary of the Exact Analysis

In addition to restrictive assumptions in the noise model (constant noise level, gaussian noise) our analysis considered the local performance only. Local performance measures are essential because evolution strategies operate locally. The probability to generate an offspring that is better than its parent decreases if the mutation rate is increased. Rechenberg (1973) introduced the concept of the *evolution window*: It states that evolutionary progress occurs only within a very narrow band of the mutation step size σ.

Although local performance measures may be useful to predict the global performance, e.g. for the sphere model, the situation in real-world scenarios may be different. The chance of finding an absolute optimum of f among several local minima depends on the structure of its fitness landscape.

The analytical approximation requires the identification of relevant factors as in the experimental analysis, consider for example the selection phase for regression trees or the forward selection mechanisms for regression models. Beyer (2001) notes that this selection process is the "source for a more in-depth understanding of the functioning of ES."

Regarding the influence of threshold selection on the global performance, we will use the statistical selection methods introduced in previous chapters of this thesis.

7.4 Global Performance

The first hypothesis to be tested reads:

(H-7.6) Threshold selection has no influence on the global performance of an evolution strategy, if the function values are disturbed by additive, gaussian noise.

It is not difficult to find a problem and an algorithm design to reject (H-7.6). For example, the test function `sphere` and the $(1+1)$-ES have been chosen to demonstrate the effect of threshold acceptance. See Table 7.4 for the problem design and Table 7.5 for the algorithm design. Further experiments (not shown here) support the assumption that threshold can also influence the global behavior of the algorithm. Negative threshold values, for example $\tau = -2$, result in an improved performance (MBST) of the $(1+1)$-ES. Other strictly convex functions, such as the `abs` function, show a similar behavior.

Until now, we have analyzed known test functions on simple test problems. Scientific methods are needed to transfer the results from the simplified models to complex ones. Recall the discussion of the role of models in science in Section 1.5: We claim that statistical

Table 7.4: *Problem design for the* $(1+1)$ *runs.*

Design	n	$t_{\max}$	d	Init.	Term.	$x^{(0)}$	Perf.	Noise
$x^{(3)}_{\texttt{sphere}}$	10	1000	2	DETEQ	EXH	100.0	MBST	$0-2$

testing (NPT*) in combination with the sequential parameter optimization from Chapter 6 provides suitable means to analyze algorithms. Our goal in this analysis is to determine how much variance in the algorithm's performance is explained by its parts or factors, i.e. by threshold selection. If a simulation study reveals that the new operator has an influence on the algorithm's performance, various algorithm designs can be analyzed statistically to determine a suitable design for a given problem.

The TS-scheme is a good example for a problem-dependent operator. Results from the sphere function are not directly transferable to real-world problems, because the determination of an adequate threshold value depends on the curvature. In addition, we can note that assumptions on the noise distributions that are usually required to derive theoretical results are met only approximately in practice.
Before the experimental study is conducted, we present some known results.

Threshold Rejection and Elevator Group Control

Beielstein et al. (2003a) report that threshold rejection was successfully integrated in an evolution strategy to optimize an elevator group controller. During the first phase of the optimization process a relatively high error of the first kind (erroneously accepting a worse candidate) is used. This α error is reduced during the optimization process. The α value is inversely proportional to the threshold value τ, as can be seen from Equation 7.17, the threshold value is increased while approaching the optimum. This idea can be formulated as a heuristic, see Figure 7.12. This heuristic should provide the opportunity to escape from local optima and is implemented in many algorithms, for example in simulated annealing.

Integrating this annealing schedule into an evolution strategy might improve its performance. The determination of a threshold value can be performed in three steps.

1. The error of the first kind is updated. An annealing schedule for the error of the first kind α can be implemented as follows:

$$\alpha(t) = 1/2\left(1 - (t/t_{\max})^2\right),$$

with $1 \leq t \leq t_{\max}$ the number of function evaluations. In the beginning, α is close to 0.5. This enables the selection of worse candidate solutions. Reducing the α value during the optimization process results in a smaller error probability of selecting the wrong candidate.

2. The noise level is estimated. The pooled estimate of the variance s_ν^2 can be calculated as in Equation 7.6.

Annealing Schedule During the initial iterations of a search algorithm it is advantageous to accept worse candidate solutions. The probability of accepting a worse candidate should be continuously reduced as the number of iterations increases.

Figure 7.12: *Heuristic. This annealing schedule can be found in many search heuristics.*

3. Based on the α value and on the estimate of the noise level s_ν^2, the threshold value is calculated as
$$\tau(\alpha(t)) = t_{\alpha(t),r-1} s_\nu, \tag{7.32}$$
where r denotes the sample size. Note that $\tau(t)$ goes to $+\infty$ as t approaches $t_{\max}$. This formula is purely heuristic. It combines information from the search process and information from the noise level. Although it reflects some theoretical results, it was not derived theoretically.

Beielstein et al. (2003a) suggest that the threshold value τ is relatively robust to changes in the noise level. The annealing schedule from Figure 7.12 uses a positive threshold value that increases as the number of function values grows. We will analyze schemes with increasing and decreasing threshold values.

7.4.1 A Simple Threshold Selection Scheme for Evolution Strategies

Consider a situation where the function values are disturbed by constant noise. Then, the variance can be determined only once at the beginning of the optimization task to adjust the threshold value. A strictly increasing, or decreasing, threshold value during the search process can improve the algorithm's performance.

The $(1+1)$-ES with algorithm design $x_{1+1}^{(0)}$ from Table 7.5 was chosen for this analysis. The problem design $x_{\texttt{sphere}}^{(4)}$ from Table 7.6 was used to determine the maximum number of function evaluations for the comparisons. The success limit was set to 10^{-4}. An analysis of the run length distributions reveals that a budget of $t_{\max} = 500$ function evaluations is sufficient: 100% of the runs were able to reach this limit with $t_{\max}$ function evaluations.

The $(1+1)$-ES for noisy functions requires the specification of the following parameters: Adaptation interval s_n, success rate s_r, step size adjustment factor s_a, starting value of the step size σ, and the number of function reevaluations r. In addition to these parameters, the $(1+1)$-TS strategy requires the specification of a threshold parameter τ. The continuous update rule ($s_{1/5} = \text{cont}$) was used and the initial step size $\sigma^{(0)}$ was held constant during the experiments: $\sigma^{(0)} = 1$. Hence, there are 5 remaining parameters.

The sequential parameter optimization was used to tune the $(1+1)$-ES first. As this procedure was detailed in Chapter 6, we only present results in the following. The resulting algorithm design is shown in Table 7.5. These results correspond to the theoretical results: Under noise, the 1/5 success rule has to be modified. The SPO procedure suggests a 1/40 success rule. A negative threshold value $\tau = -0.41$ is considered advantageous.

The following four algorithms were subject to our analysis:

(ES) The standard $(1+1)$-ES. It will be referred to as $x_{1+1}^{(0)}$ in Table 7.7.

(TS) The tuned (1+1)-TS. This variant uses a constant threshold value that was determined with the SPO procedure. It will be referred to as x_{TS}^*.

Design	s_n	s_r	s_a	τ	$\sigma^{(0)}$	r
$x_{1+1}^{(0)}$	100	5	0.85	0.0	1.0	1
x_{TS}^*	76	39.39	0.74	−0.41	1.0	1

Table 7.5: *(1+1)-ES: Algorithm design.*

Table 7.6: *Problem designs for the final study. The design* $x^{(4)}_{\texttt{sphere}}$ *was chosen to determine* $t_{\max}$*, the maximum number of function evaluations. The design* $x^{(5)}_{\texttt{sphere}}$ *has been used to perform the comparisons.*

Design	n	$t_{\max}$	d	Init.	Term.	$x^{(0)}$	Perf.	Noise
$x^{(4)}_{\texttt{sphere}}$	50	500	1	DETEQ	FSOL	100	SCR	0
$x^{(5)}_{\texttt{sphere}}$	500	500	1	DETEQ	EXH	100	MBST	10

(TRD) A $(1+1)$-TS that uses an adaptive threshold scheme. Increase τ according to

$$\tau(t) = \left(\frac{t}{t_{\max}}\right)\sigma^2_\epsilon. \tag{7.33}$$

This variant will be referred to as x^*_{TRD}.

(TAD) A $(1+1)$-TS that uses an adaptive threshold scheme. Increase τ according to

$$\tau(t) = \left(-1 + \frac{t}{t_{\max}}\right)\sigma^2_\epsilon. \tag{7.34}$$

This variant will be referred to as x^*_{TAD}.

Results from the experiments are shown in Table 7.7. Figure 7.13 presents a plot of the observed significance level versus the hypothesized difference in means of the standard (1+1)-ES and the TA algorithm. The observed differences are $\delta = 11.44$, 9.36, and 5.59 for $n =$ 10, 50, and 500 experiments respectively (mean function value after optimizing the `sphere` function with a budget of 500 function evaluations, noise level $\sigma^2_\epsilon = 10$). Consider $n = 50$: A difference as small as 3.6, that would occur frequently, has an observed significance level smaller than 0.1. This is a strong indication that the observed difference is larger than 3.6. This difference can already be detected with a small sample size.

Now that we have presented a first study to illustrate our approach, one might ask how to continue. Threshold selection analyzed so far was applied to $k = 2$ candidate solutions only. Recalling the survey of different selection schemes in the beginning of this chapter, one can see that threshold selection is a basic selection method. It can easily be extended to the single stage procedure from Figure 7.1 and integrated into a $(1+\lambda)$-ES or into a PSO. However, the corresponding analyses will become increasingly more complicated. Simple heuristics are more adequate in many situations. Optimization practitioners observe that the largest improvements occur during the first optimization stages. This idea leads to the concept of bounded rationality, which can be seen as an inspiring source to create smart heuristics.

Table 7.7: *Problem design* $x^{(5)}_{\texttt{sphere}}$*. Results from the optimization of the noisy* `sphere`*. Default and tuned algorithm designs.*

Design	Mean	Median	StD	Min	Max
$x^{(0)}_{1+1}$	8.58	5.08	10.73	1.0×10^{-4}	101.48
x^*_{TA}	5.35	2.53	8.88	7.41×10^{-6}	134.01
x^*_{TRD}	9.18	5.53	11.55	1.0×10^{-4}	137.1
x^*_{TAD}	3.09	1.22	4.94	4.1×10^{-6}	39.84

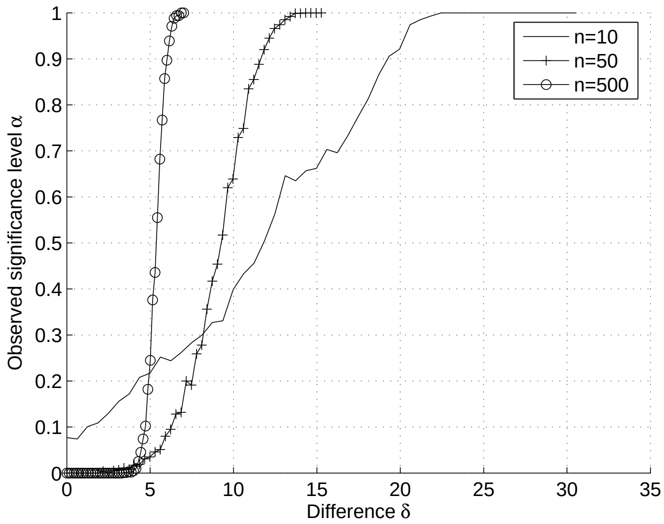


Figure 7.13: *Problem design* $x^{(5)}_{\texttt{sphere}}$. *Observed significance level. The evolution strategy with the default parameters* $x^{(0)}_{1+1}$ *is compared to the tuned version* x^*_{TAD} *that uses an adaptive threshold acceptance strategy . The curves are a strong indication for the assumption that the tuned version performs better than the default ES. This assumption is supported by the curve that represents* $n = 10$ *repeats only. This is unambiguously case RE-1 as defined in Section 1.6.*

7.5 Bounded Rationality

Simon's (1955) concept of *bounded rationality* considers

1. Cognitive limits of actual humans.

2. Environments that permit simplifications of rational decision making.

Gigerenzer and Selten (2002) note that optimization is often based on uncertain assumptions (guesswork), and there maybe about as many different outcomes of optimization strategies as there are sets of assumptions: "In these real world cases, it is possible that simple and robust heuristics can match or even outperform a specific optimization strategy." Imitation, equal weighting, take the best, take the first, and small-sample inferences are examples of fast and frugal heuristics (Goldstein et al., 2002). Another example, where a simple model outperforms a complex model, is given by Forster and Sober (1994).

Example 7.1 (Overfitting)
Curve fitting in classical and modern regression analysis consists of two steps. A family of curves is selected first, i.e. linear, quadratic, or more sophisticated functions. Simple curves are preferred in this step, consider the situation depicted in Figure 7.14. In a second step the curve in that family that fits the data best is selected. To perform the second step some measure of goodness-of-fit is necessary.

Simplicity and goodness-of-fit are two conflicting goals. Therefore, the following question arises: Why should the simplicity of a curve have any relevance to our opinions about which curve is true? Including the prediction error to these considerations provides a deeper understanding. A result in statistics from Akaike shows how simplicity and goodness-of-fit contribute to a curve's expected accuracy in making predictions. The predictive power of the curve is more important than its fit of the actual data. Curves that fit a given data set perfectly will usually perform poorly when they are used to make predictions about new data sets, a phenomenon known as overfitting. ■

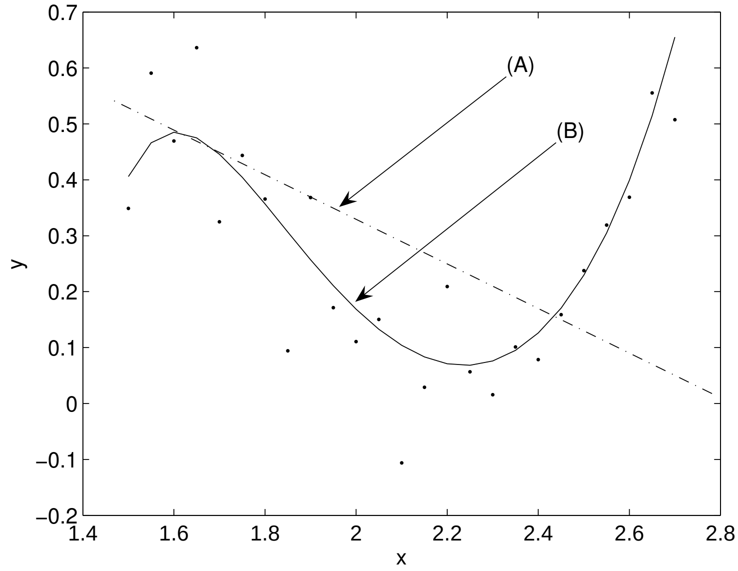


Figure 7.14: *Simplicity of curves. Linear (A) and cubic (B) curves. Imagine a third curve (C) that fits every data point. Why do scientists prefer curve (B)?*

Threshold selection, the (1 + 1)-ES, and the 1/5 success rule can be classified as simple and robust heuristics. They avoid overfitting, because they only use a minimal amount of information from the environment. Under this perspective, algorithm tuning as introduced in Chapter 6 can be seen as an analogue to curve fitting (Chapter 2): Algorithms with more exogenous strategy parameters enable a greater flexibility for the cost of an expensive tuning procedure. Worse designed algorithms can cause "overfitting"—they are able to solve one specific problem only. Domain specific heuristics require a moderate amount of information from the environment.

Does the environment permit a reduction of the required amount of information for decision making or optimization? Although simple heuristics often work well, they can be mislead easily if they are used in unsuitable environments. Consider for example the 1/5 rule: Its validity is not restricted to the `sphere` function. However, Beyer (2001) describes fitness landscapes in which the 1/5 rule fails, for example when the objective function is not continuously differentiable in the neighborhood of the parental position vector. Recognizing the situations in which domain specific heuristics perform better than other strategies provides a better understanding of their mechanisms. Understanding is seen here as to figure out in which environments a simple tool can match or even outperform more complex tools. Gigerenzer et al. (1999) use the term "ecological rationality" for this concept.

7.6 Summary

The basic ideas from this chapter can be summarized as follows:

1. Indifference zone approaches require the specification of the distance δ^* and the probability of a correct selection P^*. The IZ procedure assumes known and common variances.

2. Subset selection requires the specification of the number of samples r and the probability of a correct selection P^*.

3. Threshold selection requires the specification of the number of samples r and the probability of a correct selection P^*.

4. The probability of a correct acceptance P^* in TS is related to the error of the first kind in hypothesis testing.

5. An annealing schedule can be used to adapt the probability of a correct acceptance during the search process of an optimization algorithm.

6. Threshold selection can be interpreted as a rule of inductive behavior, or as an automatic testing rule.

7. The 1/5 rule has to be modified if the function values are disturbed by noise.

8. Threshold selection (TS) can be characterized as a fast and frugal heuristic for decision making under uncertainty. Obviously, TS does not work in every environment.

Chapter 8

Summary and Outlook

> It is good to have an end to journey toward; but it is the journey that matters, in the end.
>
> Ernest Hemingway

Now that we have reached the end of an exploratory tour during which we discussed a broad spectrum of ideas from computer science, philosophy of science, and statistics, it is time to summarize the basic achievements.

To compare different objects is a basic human activity. Decisions are based on comparisons. However, the accuracy of the observed data that are necessary for comparisons is limited in many real-world situations.

Statistics provides a means to cope with this uncertainty or "noise". Although computer science is built on deterministic grounds, it can be advantageous to introduce uncertainty or randomness. One famous example is the quick-sort algorithm, where randomness is introduced to implement the selection procedure. Another important field is stochastic search algorithms that started their triumphal procession in the 1960s. Nowadays, stochastic search algorithms belong to the standard repertoire of every optimization practitioner who has to solve harder ones than just toy problems. However, their enormous flexibility complicates their analysis. The approach presented in this thesis suggests to treat program runs as experiments to enable a statistical analysis.

The New Experimentalists

Experiments have a long history in science, their role changed drastically during the last centuries. They have been downplayed by Aristotelians who favored deduction from first principles for a long time. But the scientific revolution of the seventeenth century declared the experimental method as the "royal road to knowledge." The first scientific journals that presented experimental results and deductions from experiments were established at that time— a completely different situation to the theoretically oriented contents of modern scientific journals. Hacking (1983), one of the most influential contemporary philosophers of science, proposes to "initiate a Back-to-Bacon movement, in which we attend more seriously to experimental science." His slogan "experiment may have a life of its own" points to several new experimentalists themes, but it does not claim that experimental work could exist independently of theory. "That would be the blind work of those whom Bacon mocked as 'mere empirics'. It remains the case, however, that much truly fundamental research precedes any relevant theory whatsoever" (Hacking, 1983).

Ian Hacking, as well as Robert Ackermann (1989), Nancy Cartwright (1983; 2000), Allen Franklin (1990), Peter Galison (1987), Ronald Giere (1999), and Deborah Mayo (1996), belongs to a group of philosophers of science who share the thesis that "focusing on aspects of experiments holds the key to avoiding or solving a number of problems, problems thought to stem from the tendency to view science from theory-dominated stances". Ackermann (1989) introduced the term "new experimentalists". One major goal of the new experimentalists is to develop statistical tools for generating reliable data from experiments and using such data to learn from experiments. Mayo (1996) proposes a modern theory of statistical testing and learning from error. This thesis is an attempt to establish a modern theory of statistical testing in computer science, especially in evolutionary computation. The *new experimentalism in evolutionary computation* has its roots in the actual debate over the epistemology of experimentation in philosophy. Based on the ideas presented by the new experimentalists, in particular on Mayo's *learning from error* and her concept of *severity*, a methodology for performing and analyzing computer experiments has been developed. By controlling the errors that occur during experimentation, we can gain insight into the dependencies and interactions of important factors.

The new experimentalists extend Popper's position that only hypotheses that are in principle falsifiable by experience should count as scientific. The resulting consequences from this position have been widely discussed in the last decades, we mention only one problem that arises from the popperian view: Hypotheses require assumptions. A serious problem arises when we have to decide whether the hypothesis itself or the supporting assumption is wrong. Moreover, these assumptions require additional assumptions, which leads to an infinite regress. And finally, there are unquestionable truths, e.g. "$1 + 1 = 2$".

Learning From Error

This thesis discusses various ways to pose the right questions, to measure the performance of algorithms, and to analyze the results. However, the statistical analysis is only the first part of the investigation, it is the beginning, and not the end. We learn about algorithms by being perspicacious investigators knowing how to produce errors. Actively generating errors is a major step forward in understanding how algorithms work. This thesis points out various sources of error in the context of evolutionary computation. Errors (ER) can be caused by the selection of an inadequate test function, erroneously specified experimental designs, wrongly specified experimental goals, inadequately selected performance measures, misinterpretations of the experimental or the statistical results. This thesis provides means to master these problems in the following ways:

(ER-1) Selection of an inadequate test function: Nowadays it is a well-accepted fact that there is no computer algorithm that performs better than any other algorithm in all cases (Droste et al., 2000). However, the interaction between the problem (environment, resources) and the algorithm is crucial for its performance. To demonstrate an effect, a test problem that is well-suited to the solver (algorithm) must be chosen.

Ceiling effects can make results from computer experiments useless. They occur when every algorithm achieves the maximum level of performance—the results are indistinguishable. *Floor effects* arise when the problem is too hard, no algorithm can produce a satisfactory solution and every statistical analysis will detect no difference in the performance. Statistical methods such as run-length distributions have been proposed to tackle this issue.

We developed an elevator simulation model (S-ring) that can be used to generate test problem instances. The results are of practical relevance. They are based on an intensive cooperation with one of the world's leading elevator manufacturer (Markon et al., 2001; Beielstein and Markon, 2002; Beielstein et al., 2003a,b; Bartz-Beielstein et al., 2003c; Bartz-Beielstein and Markon, 2004; Bartz-Beielstein et al., 2005b). The S-ring model and test functions have been discussed in Chapter 3.

(ER-2) Erroneously specified experimental designs: This source of error comprehends wrongly selected exogenous strategy parameters as well as problem parameters. *Designs* play a key role in this context: The concept of problem and algorithm designs is consequently realized. Experimental designs are used to vary and control these errors systematically. The experimenter can screen out less important factors and concentrate the analysis on the relevant ones. To give an example: Evolutionary algorithms produce random results. The experimenter can vary the input parameters of the algorithm, e.g. change the recombination operator. This leads directly to the central question "how much variance in the response is explained by the variation in the algorithm?" Statistical tools that are based on the analysis of variance (ANOVA) methodology can be used to tackle this question. Classical ANOVA, modern regression techniques like tree based regression or design and analysis of computer experiments (DACE) follow this principle. Another basic tool to perform a statistical analysis is *hypothesis testing*.

We are not the first who use design of experiments techniques to analyze algorithms. However, the first attempts to apply design of experiments techniques to evolution strategies and particle swarm optimization have been presented in Beielstein et al. (2001) and Beielstein and Markon (2001). We have developed the *sequential parameter optimization* (SPO) method: SPO combines methods from classical DOE, computational statistics, and design and analysis of computer experiments. Results from the SPO can be analyzed with NPT* tools: The experimenter can learn from errors while improving an algorithm, see also (ER-5) and (ER-6). We consider the NPT* analysis as the crucial step in the analysis of computer algorithms. Experimental designs have been introduced in Chapter 4.

(ER-3) Wrongly specified experimental goals: Gary Klein (2002) uses the term "fiction of optimization" to characterize this problem. Boundary conditions that are necessary to perform optimization tasks have been discussed in Section 6.2. Specifying and analyzing boundary conditions is in accordance with Mayo's concept of learning from error and one important step of the SPO approach. Experimental goals have been discussed in Chapter 6.

(ER-4) Inadequately selected performance measures: We distinguish different performance measures to analyze algorithms, e.g. efficiency and effectivity. This classification is based on ideas that have been presented nearly three decades ago in Schwefel (1977). This problem has been addressed in Chapter 6.

(ER-5) Misinterpretations of the experimental results: Guidelines from *experimental algorithmics* (an influential discipline from computer science) recommend to state a clear set of objectives, or to formulate a question or a hypothesis. Analysis of variance methods as well as regression methods and hypothesis testing have been presented in Chapters 2 and 6.

(ER-6) Misinterpretations of the statistical results: Serious problems arise when statistical significance and scientific meaning are not distinguished. Introducing different models provides statistical tools to deal with this problem. Based on NPT*, Mayo's extension of the classical Neyman-Pearson theory of statistical testing, we developed statistical tools that allow the objective comparison of experimental results. Misconstruals can occur, if statistical tests are not severe. Consider for example the first misconstrual (MC-1) from Section 1.6 that can be accomplished by increasing the sample size n or by reducing the significance level:

> A test can be specified that will produce a result that exceeds a pre-specified difference by the required difference. As a consequence, the null hypothesis H is rejected, even if the true difference exceeds the pre-specified difference by as little as one likes.

We developed plots of the observed significance level (OSL) as key elements for an extended understanding of the significance of statistical results. They are easy to interpret, and combine information about the p-value, the sample size, and the experimental error.
We developed a bootstrap procedure to generate OSL plots independently from any assumptions on the underlying distribution. Misinterpretations of the statistical results have been discussed in Chapters 2 and 7.

Theory and Experiment

This thesis does not solely transfer concepts to compare and improve algorithms from statistics to computer science. It presents a self-contained experimental methodology that bridges the gap between theory and experiment. The advantage of applying results from theory, for example Beyer (2001), to real world optimization problems can be analyzed in an objective manner. However, as a consequence of our considerations, the interpretation of the scientific import of these results requires human experience, or the "experimenter's skill".

Why is the experimenter's skill central in our argumentation? The experimenter's skill comprises the ability to get the apparatus to indicate phenomena in a certain way. Numerous examples from the history of science can be listed in which the invention of a new apparatus enables the experimenter to perform another investigation. Results from these experiments defined the route, which the theoreticians must follow. Gigerenzer's (2003) tool-to-theory approach extends this idea from technical apparatus to abstract entities such as statistical procedures. We summarize an example presented in Hacking (1983) to illustrate our argumentation:

Example 8.1 (The Faraday Effect)
The Faraday effect, or magneto-optical effect, describes the rotation of the plane of polarization (plane of vibration) of a light beam by a magnetic field (Encyclopaedia Britannica Online, 2001). Being a deeply religious man, Michael Faraday (1791–1867) was convinced that all forces in nature must be connected. At that time the newtonian unity of science was in confusion due to several important discoveries, i.e. the wave theory of light. Faraday unsuccessfully tried to establish a connection between electrification and light in 1822, in 1834, and in 1844. In 1845 he gave up and tried to discover a connection between the forces of electromagnetism and light. Using a special kind of dense glass, which had been developed earlier in a different context, he discovered the magneto-optical effect. Faraday had no theory of what he found. One year later, G.B. Airy integrated the experimental observations into the wave theory of light simply by adding some ad hoc *further terms to the corresponding equations. "This is a standard move in physics. In order to make the equations fit the phenomena, you pull from the shelf some fairly standard extra terms for the equations, without knowing why one rather than another will do the trick." Only 47 years later, in 1892, H.A. Lorentz combined models proposed by Kelvin and adapted by Maxwell with his electron theory.* ■

This example nicely illustrates several levels of theory. Theory, as mentioned earlier, can be characterized as *speculation*: It can be seen as the process of restructuring thoughts or playing with ideas that are based on a qualitative understanding of some general features from reality. However, there is no direct link between theory and experiment. Most initial thoughts are not directly testable. Here comes *calculation* into play. Calculation is the mathematical formulation to bring speculative thoughts into accordance with the world and to conduct an experimental verification. Calculation is the first part to bridge the gap between theory and experiment.

We have not left the classical, hypothetico-deductive grounds so far. However, to bring theory in accordance with reality is not simply a matter of calculation. To do this requires more than just quantifying speculative thoughts. The idea of beginning with speculations that are gradually cast into a form from whence experimental tests can be deduced, appears to be attractive—but it is incomplete. A very extensive activity is necessary: Model-building. Models can be all sorts of things (recall the discussion in Section 1.5). It is crucial for our reasoning that they can co-exist in theory. Despite of the common understanding that at most one model can be true, several models of the physical world can be used indifferently and interchangeably in the theoretical context. Hacking presents a typical sentence from a physics textbook as an illustrative example:

> For free particles, however, we may take either the advanced or retarded potentials, or we may put the results in a symmetrical form, without affecting the result (Mott and Sneddon, 1948).

Hence, models are not merely intermediaries that connect some abstract aspects of real phenomena by simplifying mathematical structures to theories that govern the phenomena. Why can physicists use a number of mutually inconsistent models within the same theory? Recalling the ideas presented in Chapter 1, we can state that models are the central elements of science. Models are more robust than theory, that is "you keep the model and dump the theory." The number of models scientists use in their daily routine increases from year to year. Maybe there will be one unified theory of all in some years—but "that will leave most physics intact, for we shall have to do applied physics, working out what happens from case to case (Hawking, 1980)."

Approximations appear to be a solution to bridge the gap between models for theory and models for reality. But the number of possible approximations is endless, and the correct approximation cannot be derived from theory. Going one step further, Nancy Cartwright claims that "theory itself has no truth in it" (Cartwright, 1983, 2000). We follow Hacking who gives a descriptive characterization of the interplay between theories, models, and reality:

> I prefer myself an Argentine fantasy. God did not write a Book of Nature of the sort that the old Europeans imagined. He wrote a Borgesian library, each book of which is as brief as possible, yet each book of which is inconsistent with each other. No book is redundant. For every book, there is some humanly accessible bit of Nature such that that book, and no other, makes possible the comprehension, prediction and influencing what is going on. Far from being untidy, this is the New World Leibnizianism. Leibniz said that God chose a world which maximized the variety of phenomena while choosing the simplest laws. Exactly so: but the best way to maximize phenomena and have the simplest laws is to have the laws inconsistent with each other, each applying to this or that but none applying to all.

The methodology presented in this thesis may be the missing-link needed by the practitioner to consciously apply theoretical results to practical problems—and by the theoretician to explore new ideas and to confront speculations with reality. Figure 8.1 illustrates a modified view of the relationship between theory and experiment from Chapter 1.

Outlook

The items discussed in this thesis suggest various routes for further research. We will list some of them.

The experimental approach presented in this thesis may lay the cornerstone for a "borgesian library" in evolutionary computation. Consider a theory T_1, e.g. entitled "Evolutionary Algorithms in Theory", and a theory T_2, entitled "The Theory of Evolution Strategies", see Figure 8.1. Both theories use models as tools for representing parts of the world (or the theory) for specific purposes. We distinguished representational models, that represent a certain part of the world, from instantial models, that are used to present abstract entities. Performing experiments, data can be generated to test the fit of the model with some part of the world or with some theory. As models are limited *per definitionem* they may contain laws that are inconsistent with each other. And, not only models for different theories may contain conflicting laws—even models that are used within one theory might lead to different conclusions. At this point the approach presented in this thesis becomes relevant: The experimenter can use statistical tools to investigate the error probabilities by "actively probing, manipulating, and simulating patterns of error, and by deliberately introducing known patterns of error into the collection and analysis of data"(Mayo, 1996). Consider two scenarios:

1. Experimental designs (Chapter 4) provide means to specify the essential conditions in an objective manner. Optimization practitioners can consult a "book from the problem design section of the library" and look up a candidate algorithm that might be able to solve their problem. This algorithm will not be used directly with some default parameter settings—it will be tuned before the optimization run is performed.

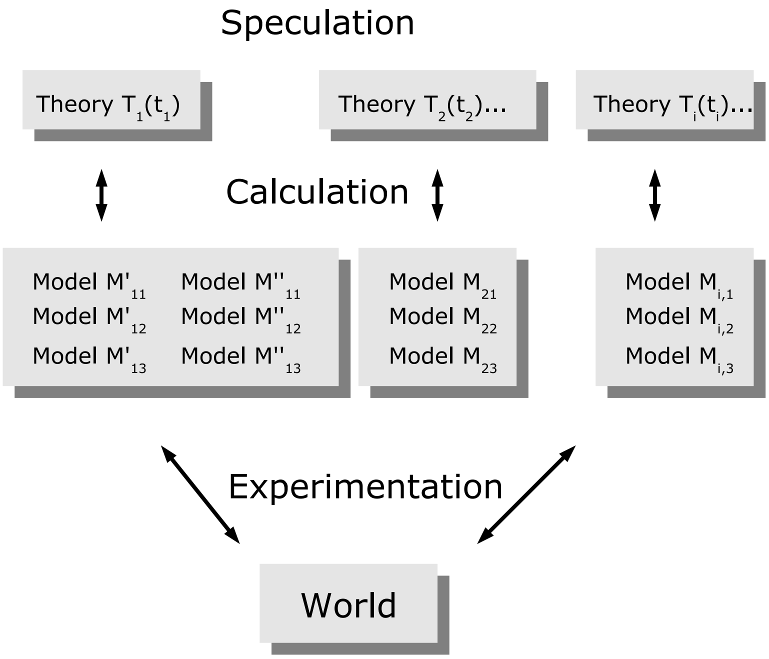


Figure 8.1: *A second attempt to model the relationship between theory and practice. The first attempt (Figure 1.1) is reconsidered. Different theories and models, even with conflicting laws, coexist. The variables t_i denote the time-dependency of some theories. Speculation can be interpreted as "playing with ideas", calculation brings speculative thoughts in accordance with models, and experimentation tests the fit of models with the world.*

2. Researchers will base the comparison of different algorithms not on their default parameterizations, but on the tuned versions. The SPO (or a similar method) enables an algorithmical tuning process with traceable costs.

Consequently, a *tool box* with many different algorithms, e.g. as suggested by Schwefel (1995), "might always be the 'optimum optimorum' for the practitioner." The methods presented in this thesis might give some valuable advice for the selection of an appropriate tool.

Recent discussions indicated a great demand for an *automated version* of the sequential parameter optimization procedure (Chapter 6). The development of such an automated tool is—at least from our perspective—a conflicting goal, because the user does not "see" what happens during the design optimization. However, SPO will gain acceptance and influence, if we keep the complexity that is necessary to apply SPO as low as possible. A first implementation of an automated SPO is under development (Bartz-Beielstein et al., 2005a).

Including the relationship between step-size adaptation and threshold selection into the analysis from the case study in Chapter 7 will provide interesting insights into the behavior of evolution strategies. In general, an analysis of the *self-adaptation* mechanisms seems to be one of the most exciting tasks for further research. Recall, that a careful examination is required to perform this task, because the evolution strategy presented in Chapter 5 required the specification of nine design variables and various interactions have to be considered.

Only first attempts have been made to apply SPO to *multi-criteria optimization* problems (Bartz-Beielstein et al., 2003b; Mehnen et al., 2004a). A discussion—similar to the one presented in Chapter 6—of performance measures for multi-criteria optimization has to be done in advance. This issue is analyzed in the collaborative research center "Design and Management of Complex Technical Processes and Systems by Means of Computational Intelligence Methods" (Beielstein et al., 2003c).

Visual tools that enable an intuitive understanding of experimental results and their scientific meaning should be developed. The observed significance level plots (Chapter 1) are merely a first step in this direction.

Based on considerations related to the concept of *bounded rationality*, one can ask in which environment an algorithm performs well (Gigerenzer et al., 1999). Not merely organic evolution, but also social evolution, might give valuable hints to develop new strategies or to understand existing behaviors. This process can be beneficial in both directions: Evolutionary algorithms can be used to model social behavior and *vice versa*. Consider for example the model of urban growth by cellular automata from Bäck et al. (1996), or a current diploma thesis that models farm size and market power on agricultural land markets with particle swarm optimization (de Vegt, 2005). We close this thesis with an analogy to depict the relativity of good algorithms (or strategies) and to demonstrate the usefulness of experience:

> In a remote stream in Alaska, a rainbow trout spies a colorful dimple on the undersurface of the water with an insect resting on top of it. Darting over with the mouth agape, the fish bites down and turns in search for its next victim. It does not get far, however, before the "insect" strikes back. The trout is yanked from the quiet stream by the whiplike pull of a fly fisherman's rod. In a world without fisherman, striking all the glitter is adaptive; it increases the chance for survival. In a world with predators, however, this once-adaptive strategy can turn a feeding fish into a fisherman's food (Goldsman et al., 2002).

Bibliography

Ackermann, R. (1989). The new experimentalism. *Brit. J. Phil. Sci.*, 40:185–190.

Anderson, R. (1997). 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*, pages 191–196. American Mathematical Society.

Armitage, P., McPherson, C., and Rowe, B. C. (1969). Repeated significance tests on accumulating data. *Journal of the Royal Statistical Society: Series A*, 132:235–244.

Arnold, D. V. (2001). Evolution strategies in noisy environments — A survey of existing work. In Kallel, L., Naudts, B., and Rogers, A., editors, *Theoretical Aspects of Evolutionary Computing*, Natural Computing, pages 239–249. Springer, Berlin.

Arnold, D. V. and Beyer, H.-G. (2003). A comparison of evolution strategies with other direct search methods in the presence of noise. *Computational Optimization and Applications*, 24(1):135–159.

Aslett, R., Buck, R. J., Duvall, S. G., Sacks, J., and Welch, W. J. (1998). Circuit optimization via sequential computer experiments: design of an output buffer. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 47(1):31–48.

Athen, H. and Bruhn, J., editors (1980). *Lexikon der Schulmathematik. Studienausgabe.* Aulis, Köln.

Azadivar, F. (1999). Simulation optimization methodologies. In Farrington, P., Nembhard, D. T., Sturrock, D. T., and Evans, G. W., editors, *Proceedings of the 1999 Winter Simulation Conference*, pages 93–100, Piscataway, New Jersey. IEEE.

Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York.

Bäck, T., Dörnemann, H., Hammel, U., and Frankhauser, P. (1996). Modeling urban growth by cellular automata. In Voigt, H.-M., Ebeling, W., Rechenberg, I., and Schwefel, H.-P., editors, *Proc. Parallel Problem Solving from Nature – PPSN IV, Berlin*, pages 636–645, Berlin. Springer.

Bandler, J., Cheng, Q., Dakroury, S., Mohamed, A., Bakr, M., Madsen, K., and Søndergaard, J. (2004). Space mapping: the state of the art. *IEEE Transactions on Microwave Theory and Techniques*, 52(1):337–361.

Banks, J., Carson, J. S., Nelson, B. L., and Nicol, D. M. (2001). *Discrete Event System Simulation*. Prentice Hall.

Barney, G. (1986). *Elevator Traffic Analysis, Design and Control*. Cambridge U.P.

Barr, R., Golden, B., Kelly, J., Rescende, M., and Stewart, W. (1995). Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, 1(1):9–32.

Barr, R. and Hickman, B. (1993). Reporting computational experiments with parallel algorithms: Issues, measures, and experts' opinions. *ORSA Journal on Computing*, 5(1):2–18.

Bartz-Beielstein, T. (2003). Experimental analysis of evolution strategies – overview and comprehensive introduction. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–157/03, Universität Dortmund.

Bartz-Beielstein, T., Blum, D., Lasarczyk, C., and Preuss, M. (2005a). Sequential parameter optimzation. Submitted to the 2005 IEEE Congress on Evolutionary Computation.

Bartz-Beielstein, T., de Vegt, M., Parsopoulos, K. E., and Vrahatis, M. N. (2004a). Designing particle swarm optimization with regression trees. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–173/04, Universität Dortmund.

Bartz-Beielstein, T., Limbourg, P., Mehnen, J., Schmitt, K., Parsopoulos, K. E., and Vrahatis, M. N. (2003a). Particle swarm optimizers for Pareto optimization with enhanced archiving techniques. In Sarker, R. et al., editors, *Proc. 2003 Congress on Evolutionary Computation (CEC'03), Canberra*, volume 3, pages 1780–1787, Piscataway NJ. IEEE Press.

Bartz-Beielstein, T., Limbourg, P., Mehnen, J., Schmitt, K., Parsopoulos, K. E., and Vrahatis, M. N. (2003b). Particle Swarm Optimizers for Pareto Optimization with Enhanced Archiving Techniques. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–153/03, Universität Dortmund.

Bartz-Beielstein, T. and Markon, S. (2004). Tuning search algorithms for real-world applications: A regression tree based approach. In Greenwood, G. W., editor, *Proc. 2004 Congress on Evolutionary Computation (CEC'04), Portland OR*, volume 1, pages 1111–1118, Piscataway NJ. IEEE Press.

Bartz-Beielstein, T., Markon, S., and Preuß, M. (2003c). Algorithm based validation of a simplified elevator group controller model. In Ibaraki, T., editor, *Proc. 5th Metaheuristics Int'l Conf. (MIC'03)*, pages 06/1–06/13 (CD–ROM), Kyoto.

Bartz-Beielstein, T. and Naujoks, B. (2004). Tuning multi criteria evolutionary algorithms for airfoil design optimization. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–159/04, Universität Dortmund.

Bartz-Beielstein, T., Parsopoulos, K. E., and Vrahatis, M. N. (2004b). Analysis of particle swarm optimization using computational statistics. In (Simos and Tsitouras, 2004), pages 34–37.

Bartz-Beielstein, T., Parsopoulos, K. E., and Vrahatis, M. N. (2004c). Design and analysis of optimization algorithms using computational statistics. *Applied Numerical Analysis & Computational Mathematics (ANACM)*, 1(2):413–433.

Bartz-Beielstein, T., Preuß, M., and Markon, S. (2003d). Validation and optimization of an elevator simulation model with modern search heuristics. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–158/03, Universität Dortmund.

Bartz-Beielstein, T., Preuß, M., and Markon, S. (2005b). Validation and optimization of an elevator simulation model with modern search heuristics. In Ibaraki, T., Nonobe, K., and Yagiura, M., editors, *Metaheuristics: Progress as Real Problem Solvers*, chapter 5, pages 109–128. Kluwer, Boston MA. (in print).

Bartz-Beielstein, T., Preuß, M., and Reinholz, A. (2003e). Evolutionary algorithms for optimization practitioners. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–151/03, Universität Dortmund.

Bartz-Beielstein, T., Preuß, M., and Reinholz, A. (2003f). Evolutionary algorithms for optimization practitioners (tutorial). 5th Metaheuristics Int'l Conf. (MIC'03) Kyoto.

Bartz-Beielstein, T., Schmitt, K., Mehnen, J., Naujoks, B., and Zibold, D. (2004d). KEA – A software package for development, analysis, and application of multiple objective evolutionary algorithms. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–185/04, Universität Dortmund.

Bechhofer, R. E., Dunnett, C. W., Goldsman, D. M., and Hartmann, M. (1990). A comparison of the performances of procedures for selecting the normal population having the largest mean when populations have a common unknown variance. *Communications in Statistics*, B19:971–1006.

Bechhofer, R. E., Santner, T. J., and Goldsman, D. M. (1995). *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons*. Wiley.

Beielstein, T. (2003). Tuning evolutionary algorithms – Overview and comprehensive introduction. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–148/03, Universität Dortmund.

Beielstein, T., Dienstuhl, J., Feist, C., and Pompl, M. (2001). Circuit design using evolutionary algorithms. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–122/01, Universität Dortmund.

Beielstein, T., Dienstuhl, J., Feist, C., and Pompl, M. (2002a). Circuit design using evolutionary algorithms. In Fogel, D. B., El-Sharkawi, M. A., Yao, X., Greenwood, G., Iba, H., Marrow, P., and Shackleton, M., editors, *Proc. 2002 Congress on Evolutionary Computation (CEC'02) within Third IEEE World Congress on Computational Intelligence (WCCI'02), Honolulu HI*, pages 1904–1909, Piscataway NJ. IEEE Press.

Beielstein, T., Ewald, C.-P., and Markon, S. (2003a). Optimal elevator group control by evolution strategies. In Cantú-Paz, E., Foster, J. A., Deb, K., Davis, L. D., Roy, R., O'Reilly, U.-M., Beyer, H.-G., et al., editors, *Proc. Genetic and Evolutionary Computation Conf. (GECCO 2003), Chicago IL, Part II*, volume 2724 of *Lecture Notes in Computer Science*, pages 1963–1974, Berlin. Springer.

Beielstein, T. and Markon, S. (2001). Threshold selection, hypothesis tests, and DoE methods. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–121/01, Universität Dortmund.

Beielstein, T. and Markon, S. (2002). Threshold selection, hypothesis tests, and DOE methods. In Fogel, D. B., El-Sharkawi, M. A., Yao, X., Greenwood, G., Iba, H., Marrow, P., and Shackleton, M., editors, *Proc. 2002 Congress on Evolutionary Computation (CEC'02) within Third IEEE World Congress on Computational Intelligence (WCCI'02), Honolulu HI*, pages 777–782, Piscataway NJ. IEEE Press.

Beielstein, T., Markon, S., and Preuß, M. (2003b). A parallel approach to elevator optimization based on soft computing. In Ibaraki, T., editor, *Proc. 5th Metaheuristics Int'l Conf. (MIC'03)*, pages 07/1–07/11 (CD–ROM), Kyoto.

Beielstein, T., Mehnen, J., Schönemann, L., Schwefel, H.-P., Surmann, T., Weinert, K., and Wiesmann, D. (2003c). Design of evolutionary algorithms and applications in surface reconstruction. In Schwefel, H.-P., Wegener, I., and Weinert, K., editors, *Advances in Computational Intelligence – Theory and Practice*, pages 145–193. Springer, Berlin.

Beielstein, T., Parsopoulos, K. E., and Vrahatis, M. N. (2002b). Tuning PSO parameters through sensitivity analysis. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–124/02, Universität Dortmund.

Beielstein, T., Preuss, M., and Markon, S. (2003d). A parallel approach to elevator optimization based on soft computing. Interner Bericht des Sonderforschungsbereichs 531 *Computational Intelligence* CI–147/03, Universität Dortmund.

Belisle, C. J. P. (1992). Convergence theorems for a class of simulated annealing algorithms. *Journal Applied Probability*, 29:885–895.

Bentley, P. (2002). ISGEC workshop on standards at GECCO 2002. http://www.cs.ucl.ac.uk/staff/P.Bentley/standards.html.

Berger, J. O. (2003). Could Fisher, Jeffreys and Neyman have agreed on testing? *Statistical Science*, pages 1–32.

Beyer, H.-G. (2000). Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. *CMAME (Computer methods in applied mechanics and engineering)*, 186:239–267.

Beyer, H.-G. (2001). *The Theory of Evolution Strategies*. Natural Computing Series. Springer, Heidelberg.

Beyer, H.-G. and Schwefel, H.-P. (2002). Evolution strategies – A comprehensive introduction. *Natural Computing*, 1:3–52.

Birattari, M., Stützle, T., Paquete, L., and Varrentrapp, K. (2002). A racing algorithm for configuring metaheuristics. In et al., W. B. L., editor, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 11–18. Morgan Kaufmann.

Bohachevsky, I. (1986). Generalized simulated annealing for function optimization. *Technometrics*, 28(3):209–217.

Box, G. E. P. (1957). Evolutionary operation: A method for increasing industrial productivity. *Applied Statistics*, 6:81–101.

Box, G. E. P. and Draper, N. R. (1987). *Empirical Model Building and Response Surfaces*. Wiley.

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (1978). *Statistics for experimenters*. Wiley series in probabilty and mathematical statistics: Applied probability and statistics. Wiley.

Branke, J., Schmidt, C., and Schmeck, H. (2001). Efficient fitness estimation in noisy environments. In et al., L. S., editor, *Genetic and Evolutionary Computation Conference (GECCO'01)*. Morgan Kaufmann.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth.

Briest, P., Brockhoff, D., Degener, B., Englert, M., Gunia, C., Heering, O., Jansen, T., Leifhelm, M., Plociennik, K., Röglin, H., Schweer, A., Sudholt, D., Tannenbaum, S., and Wegener, I. (2004). Experimental supplements to the theoretical analysis of EAs on problems from combinatorial optimization. In Yao, X., Burke, E., Lozano, J. A., Smith, J., Merelo-Guervós, J., Bullinaria, J., Rowe, J., Kabán, P., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature - PPSN VIII*, volume 3242 of *LNCS*, pages 21–30, Birmingham, UK. Springer.

Bussieck, M., Drud, A., Meeraus, A., and Pruessner, A. (2003). Quality assurance and global optimization. In *Global Optimization and Constraint Satisfaction: First International Workshop on Global Constraint Optimization and Constraint Satisfaction, COCOS 2002*, Lecture Notes in Computer Science 2861/2003, pages 223–238, Berlin. Springer.

Cartwright, N. (1983). *How the Laws of Physics Lie*. Oxford University Press.

Cartwright, N. (2000). *The Dappled World: A Study of the Boundaries of Science*. Cambridge University Press, Cambridge.

Chalmers, A. F. (1999). *What is This Thing Called Science*. Universiy of Queensland Press, 3. edition.

Chambers, J., Cleveland, W., Kleiner, B., and Tukey, P. (1983). *Graphical Methods for Data Analysis*. Wadsworth.

Chambers, J. M. and Hastie, T. H., editors (1992). *Statistical Models in S*. Wadsworth & Brooks/Cole, Pacific Grove, California.

Chiarandini, M. and Stützle, T. (2002). Experimental evaluation of course timetabling algorithms. Technical Report AIDA-02-05, FG Intellektik, TU Darmstadt.

Clerc, M. and Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73.

Cohen, J. (1990). Things I have learned (so far). *American Psychologist*, 45:1304–1312.

Cohen, P., Gent, I. P., and Walsh, T. (2000). Empirical methods for AI, tutorial given at AAAI, ECAI and Tableaux conferences in 2000.

Cohen, P. R. (1995). *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge, MA.

Coleman, T. and Zhang, Y. (2004). *Optimization Toolbox for Use with MATLAB*. The MathWorks, Inc.

Colemann, D. E. and Montgomery, D. C. (1993). A systematic approach to planning for a designed industrial experiment. *Technometrics*, 35:1–27.

Collett, D. (1991). *Modelling Binary Data*. Chapman and Hall, London.

Cox, D. R. (1977). The role of significance tests. *Scandinavian Journal of Statistics*, 4:49–70.

Cox, D. R. and Hinkley, D. V. (1974). *Theoretical statistics*. Chapman and Hall, London.

Crites, R. and Barto, A. (1998). Elevator group control using multiple reinforcement learning agents. *Machine Learning*, 33(2-3):235–262.

Croarkin, C. and Tobias, P., editors (2004). *NIST/SEMATECH e-Handbook of Statistical Methods*. National Institute of Standards and Technology. http://www.itl.nist.gov/div898/handbook/ (April 2004).

de Groot, A. (1946/1978). *Thought and Choice in Chess*. Mouton, New York.

de Vegt, M. (2005). Einfluss verschiedener Parametrisierungen auf die Dynamik des Partikelschwarm Verfahrens: Eine empirische Analyse. Diplomarbeit. Fachbereich Informatik, Universität Dortmund.

Demetrescu, C. and Italiano, G. F. (2000). What do we learn from experimental algorithmics? In *Mathematical Foundations of Computer Science*, pages 36–51.

Dolan, E. D. and More, J. J. (2001). Benchmarking optimization software with performance profiles. Technical Report ANL/MCS-P861-1200, Argonne National Laboratory.

Draper, N. R. and Smith, H. (1998). *Applied regression analysis*. Wiley series in probability and statistics. Wiley, New York, 3rd edition.

Driml, M. and Hanš, O. (1967). On a randomized optimization procedure. In Kožešnik, J., editor, *Transactions of the 4th Prague Conference on Information Theory, Statistical Decision Functions and Random Processes (held at Prague 1965)*, pages 273–276, Prague. Czechoslovak Academy of Sciences.

Droste, S., Jansen, T., and Wegener, I. (2000). Optimization with randomized search heuristics: The (A)NFL theorem, realistic scenarios, and difficult functions. Technical Report CI–91/00, SFB 531, Universität Dortmund.

Dueck, G. and Scheuer, T. (1990). Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90:161–175.

Eberhart, R. and Shi, Y. (1998). Comparison between genetic algorithms and particle swarm optimization. In Porto, V., Saravanan, N., Waagen, D., and Eiben, A., editors, *Evolutionary Programming*, volume VII, pages 611–616. Springer, Berlin.

Efron, B. and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Chapman and Hall, London.

Eiben, A., Hinterding, R., and Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Trans. on Evolutionary Computation*, 3(2):124–141.

Eiben, A. and Jelasity, M. (2002). 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.

Eiben, A. and Smith, J. (2003). *Introduction to Evolutionary Computing*. Springer, Berlin.

Emmerich, M., Giotis, A., Özdemir, M., Bäck, T., and Giannakoglou, K. (2002). Metamodel-assisted evolution strategies. In et al., J. J. M. G., editor, *Parallel Problem Solving from Nature – PPSN VII, Proc. Seventh Int'l Conf., Granada*, pages 361–370, Berlin. Springer.

Encyclopaedia Britannica Online (2001). "Faraday effect". http://members.eb.com/bol/topic?eu=34314&sctn=1 (28 October 2001).

Fabian, V. (1962). On multiple decision methods for ranking population means. *Ann. Math. Stat.*, 33:248–254.

Federov, V. (1972). *Theory of optimal experiments*. Academic Press.

Feldt, R. and Nordin, P. (2000). Using factorial experiments to evaluate the effect of genetic programming parameters. In Poli, R., Banzhaf, W., Langdon, W. B., Miller, J. F., Nordin, P., and Fogarty, T. C., editors, *Genetic Programming, Proceedings of EuroGP'2000*, volume 1802 of *LNCS*, pages 271–282, Edinburgh. Springer.

Felscher, W. (1998). Two dicta. Historia Matematica Mailing List Archive. http://sunsite.utk.edu/math_archives/.http/hypermail/historia/aug98/0021.html. Article posted: Wed, 5 Aug 1998.

Fisher, R. A. (1935). *The Design of Experiments*. Oliver and Boyd, Edinburgh.

Folks, J. (1981). *Ideas of Statistic*. Wiley.

Forster, M. and Sober, E. (1994). How to tell when simpler, more unified, or less ad hoc theories will provide more accurate predictions. *Brit. J. Phil. Sci.*, 45:1–35.

François, O. and Lavergne, C. (2001). Design of evolutionary algorithms – a statistical perspective. *IEEE Transactions on Evolutionary Computation*, 5(2):129–148.

Franco, J. and Paull, M. (1983). Probabilistic analysis of the Davis Putnam procedure for solving the satisfiability problem. *Discrete Applied Mathematics*, 5(1):77–87.

Franklin, A., editor (1990). *Experiment, right or wrong*. Cambridge University Press, Cambridge.

Franklin, A. (Summer 2003). Experiment in physics. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*.

Galavotti, M. C., editor (2003). *Observation and experiment in the natural and social sciences*. Kluwer, Dordrecht.

Galison, P. (1987). *How experiments end*. University of Chicago Press, Chicago.

Gentle, J. E., Härdle, W., and Mori, Y. (2004a). Computational statistics: An introduction. In (Gentle et al., 2004b), pages 3–16.

Gentle, J. E., Härdle, W., and Mori, Y., editors (2004b). *Handbook of Computational Statistics*. Springer, Berlin.

Giere, R. (1999). Using models to represent reality. In Magnani, L., editor, *Model based reasoning in scientific discovery. Proceedings of the International Conference on Model-Based Reasoning in Scientific Discovery*, pages 41–57, New York. Kluwer.

Gigerenzer, G. (2003). Where do new ideas come from? A heuristic of discovery in cognitive sciences. In (Galavotti, 2003), pages 99–139.

Gigerenzer, G. and Selten, R., editors (2002). *Bounded Rationality – The Adaptive Toolbox*. The MIT Press, Cambridge, Massachusetts.

Gigerenzer, G., Todd, P. M., and the ABC research group (1999). *Simple heuristics that make us smart*. Oxford University Press, New York.

Giunta, A., Wojtkiewicz Jr., S., and Eldred, M. (2003). Overview of modern design of experiments methods for computational simulations. In *Proceedings of the 41st AIAA Aerospace Sciences Meeting and Exhibit*. paper AIAA-2003-0649.

Goldberg, A. (1979). On the complexity of the satisfiability problem. Technical Report 16, Courant Computer Science Report, New York University, NY.

Goldberg, A., Purdom, P. W., and Brown, C. A. (1982). Average time analyses of simplified Davis-Putnam procedures. *Inf. Process. Lett.*, 15(2):72–75.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.

Goldsman, D., Kim, S., Marshall, W. S., and Nelson, B. L. (2002). Ranking and selection for steady-state simulation: Procedures and perspectives. *INFORMS Journal on Computing*, 14:2–19.

Goldsman, D. and Nelson, B. L. (1998). Statistical screening, selection, and multiple comparison procedures in computer simulation. In *Proceedings of the 30th conference on Winter simulation*, pages 159–166. IEEE Computer Society Press.

Goldstein, D., Gigerenzer, G., Hogart, R., Kacelnik, A., Kareev, Y., Klein, G., Martignon, L., Payne, J., and Schlag, K. (2002). Group report: Why and when do simple heuristics work? In (Gigerenzer and Selten, 2002), pages 174–190.

Gooding, D., Pinch, T., and Schaffer, S. (1989). *The uses of experiment: Studies in the natural sciences*. Cambridge University Press, Cambridge.

Gregoire, T. (2001). Biometry in the 21st century: Whither statistical inference? Invited Keynote. Forest Biometry, Modelling and Information Science. Proceedings of a IUFRO 4.11 conference held at the University of Greenwich, June 2001.

Gregory, D., Gao, L.-X., Rosenberg, A., and Cohen, P. (1996). An empirical study of dynamic scheduling on rings of processors. In *8th IEEE Symp. on Parallel and Distr. Processing*, pages 470–473.

Guala, F. (2003). Experimental localism and external validity. *Philosophy of Science*, 70:1195–1205.

Gupta, S. S. (1965). On some multiple decision (selection and ranking) rules. *Technometrics*, 7:225–245.

Hacking, I. (1983). *Representing and intervening*. Cambridge University Press.

Hacking, I. (1996). *Einführung in die Philosophie der Naturwissenschaften*. Reclam.

Hacking, I. (2001). *An Introduction to Probability and Inductive Logic*. Cambridge University Press.

Hartmann, M. (1988). An improvement on Paulsson's sequential ranking procedure. *Sequential Analysis*, 7:363–372.

Hartmann, M. (1991). An improvement on Paulsson's procedure for selecting the population with the largest mean from k normal populations with a common unknown variance. *Sequential Analysis*, 10:1–16.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer, Berlin.

Hawking, S. W. (1980). *Is the end in sight for theoretical physics? : an inaugural lecture*. Cambridge University Press, Cambridge, New York.

Hillstrom, K. E. (1977). A simulation test approach to the evaluation of nonlinear optimization algorithms. *ACM Trans. Math. Softw.*, 3(4):305–315.

Hooker, J. (1994). Needed : An empirical science of algorithms. *Operations Res.*, 42(2):201–212.

Hooker, J. (1996). Testing heuristics: We have it all wrong. *Journal of Heuristics*, 1(1):33–42.

Hoos, H. H. (1998). *Stochastic Local Search – Methods, Models, Applications*. PhD thesis, Technische Universität Darmstadt.

Isaaks, E. H. and Srivastava, R. M. (1989). *An Introduction to Applied Geostatistics*. Oxford University Press.

Jansen, T. and Wegener, I. (2000). Evolutionary algorithms: How to cope with plateaus of constant fitness and when to reject strings of the same fitness. Technical Report CI–96/00, Universität Dortmund, Fachbereich Informatik.

Jarvie, I. C. (1998). Popper, Karl Raimund. In Craig, E., editor, *Routledge Encyclopedia of Philosophy*. Routledge, London. Retrieved November 19, 2003, from http://www.rep.routledge.com/article/DD052SECT2.

Jin, R., Chen, W., and Sudjitanto, A. (2002). On sequential sampling for global metamodeling in engineering design. In *Design Automation Conference*, pages 1–10. ASME.

Jin, Y. (2003). A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing Journal*. In press.

Johnson, D. S. (2002). A theoretician's guide to the experimental analysis of algorithms. In Goldwasser, M. H., Johnson, D. S., and McGeoch, C. C., editors, *Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges*, pages 215–250, Providence. American Mathematical Society.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., and Schevon, C. (1989). Optimization by simulated annealing: an experimental evaluation. Part I, graph partitioning. *Operations Research*, 37(6):865–892.

Johnson, D. S., Aragon, C. R., McGeoch, L. A., and Schevon, C. (1991). Optimization by simulated annealing: an experimental evaluation. Part II, graph coloring and number partitioning. *Operations Research*, 39(3):378–406.

Jones, D., Schonlau, M., and Welch, W. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492.

Kan, A. (1976). *Machine Scheduling Problems: Classification, Complexity and Computation*. Martinus Nijhoff, The Hague.

Kelton, W. (2000). Experimental design for simulation. In Joines, J., Barton, R., Kang, K., and Fishwick, P., editors, *Proceedings of the 2000 Winter Simulation Conference*.

Kempthorne, O. and Folks, L. (1971). *Probability, Statistics, and Data Analysis*. Iowa State University Press.

Kennedy, J. (2003). Bare bones particle swarms. In *Proc. 2003 IEEE Swarm Intelligence Symposium*, pages 80–87. IEEE Press.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proc. IEEE Int.'l Conf. on Neural Networks*, volume IV, pages 1942–1948, Piscataway, NJ. IEEE Service Center.

Kennedy, J. and Eberhart, R. (2001). *Swarm Intelligence*. Morgan Kaufmann Publishers.

Kim, S.-H. and Nelson, B. L. (2001). A fully sequential procedure for indifference-zone selection in simulation. *ACM Trans. Model. Comput. Simul.*, 11(3):251–273.

Kleijnen, J. P. C. (1987). *Statistical Tools for Simulation Practitioners*. Marcel Dekker, New York.

Kleijnen, J. P. C. (1997). Experimental design for sensitivity analysis, optimization, and validation of simulation models. In Banks, J., editor, *Handbook of simulation*. Wiley, New York.

Kleijnen, J. P. C. (2001). Experimental designs for sensitivity analysis of simulation models. In et al., A. W., editor, *Proceedings of EUROSIM 2001*.

Kleijnen, J. P. C. and Van Groenendaal, W. (1992). *Simulation - A Statistical Perspective*. Wiley, Chichester.

Klein, G. (2002). The fiction of optimization. In (Gigerenzer and Selten, 2002), pages 103–121.

Knuth, D. (1981). *The Art of Computer Programming*. Addison-Wesley, Reading, 2. edition.

Kursawe, F. (1999). *Grundlegende empirische Untersuchungen der Parameter von Evolutionsstrategien — Metastrategien*. Dissertation, Fachbereich Informatik, Universität Dortmund.

Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. (1998). Convergence properties of the Nelder–Mead simplex method in low dimensions. *SIAM J. on Optimization*, 9(1):112–147.

Law, A. and Kelton, W. (2000). *Simulation Modeling and Analysis*. McGraw-Hill, New York, 3rd edition.

Lewis, R., Torczon, V., and Trosset, M. (2000). Direct search methods: Then and now. *Journal of Computational and Applied Mathematics*, 124(1–2):191–207.

Lophaven, S., Nielsen, H., and Søndergaard, J. (2002a). Aspects of the Matlab Toolbox DACE. Technical Report IMM-REP-2002-13, Informatics and Mathematical Modelling, Technical University of Denmark.

Lophaven, S., Nielsen, H., and Søndergaard, J. (2002b). DACE - A Matlab Kriging Toolbox. Technical Report IMM-REP-2002-12, Informatics and Mathematical Modelling, Technical University of Denmark.

Mammen, E. and Nandi, S. (2004). Bootstrap and resampling. In (Gentle et al., 2004b), pages 467–495.

Markon, S. (1995). *Studies on Applications of Neural Networks in the Elevator System*. PhD thesis, Kyoto University.

Markon, S., Arnold, D. V., Bäck, T., Beielstein, T., and Beyer, H.-G. (2001). Thresholding – A selection operator for noisy ES. In Kim, J.-H., Zhang, B.-T., Fogel, G., and Kuscu, I., editors, *Proc. 2001 Congress on Evolutionary Computation (CEC'01), Seoul*, pages 465–472, Piscataway NJ. IEEE Press.

Markon, S. and Nishikawa, Y. (2002). On the analysis and optimization of dynamic cellular automata with application to elevator control. The 10th Japanese-German Seminar, Nonlinear Problems in Dynamical Systems, Theory and Applications. Noto Royal Hotel, Hakui, Ishikawa, Japan.

Martinez, W. and Martinez, A. (2002). *Computational Statistics Handbook with MATLAB*. Chapman & Hall / CRC.

Matyáš, J. (1965). Random Optimization. *Automation and Remote Control*, 26(2):244–251.

Mayo, D. G. (1981a). In defense of the Neyman-Pearson theory of confidence intervals. *Philosophy of Science*, 48:268–280.

Mayo, D. G. (1981b). Testing statistical testing. In Pitt, J., editor, *Philosophy in Economics*, pages 175–201. Reidel, Dordrecht.

Mayo, D. G. (1983). An objective theory of statistical testing. *Synthese*, 57:297–340.

Mayo, D. G. (1996). *Error and the Growth of Experimental Knowledge*. The University of Chicago Press.

Mayo, D. G. (1997). Severe tests, arguing from error, and methodological underdetermination. *Philosophical studies*, 86:243–266.

Mayo, D. G. (2003). Comment on a paper by Berger. *Statistical Science*, 18(1):19–24.

McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models*. Chapman and Hall, 2nd edition.

McGeoch, C. (1986). *Experimental analysis of algorithms*. PhD thesis, Carnegie Mellon University.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). 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.

McPherson, C. and Armitage, P. (1971). Repeated significance tests on accumulating data when the null hypothesis is not true. *Journal of the Royal Statistical Society*, A 134:15–25.

Mehnen, J., Michelitsch, T., Bartz-Beielstein, T., and Henkenjohann, N. (2004a). Systematic analyses of multi-objective evolutionary algorithms applied to real-world problems using statistical design of experiments. In Teti, R., editor, *Proc. Fourth Int'l Seminar Intelligent Computation in Manufacturing Engineering (CIRP ICME'04)*, volume 4, pages 171–178, Universität von Neapel. C.O.C. Com. Org. Conv. CIRP ICME'04.

Mehnen, J., Michelitsch, T., Bartz-Beielstein, T., and Schmitt, K. (2004b). Evolutionary optimization of mould temperature control strategies: Encoding and solving the multiobjective problem with standard evolution strategy and kit for evolutionary algorithms. *Journal of Engineering Manufacture (JEM)*, 218(B6):657–666.

Merriam-Webster Online Dictionary (2004a). "Statistics". http://www.merriam-webster.com (2 April 2004).

Merriam-Webster Online Dictionary (2004b). "Theory". http://www.merriam-webster.com (2 April 2004).

Mertens, H. (1990). *Moderne – Sprache – Mathematik: eine Geschichte des Streits um die Grundlagen der Disziplin und des Subjekts formaler Systeme*. Suhrkamp.

Metropolis, N. and Ulam, S. (1949). The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341.

Minsky, M. (1985). *The Society of Mind*. Simon and Schuster, New York.

Mitchell, D. G., Selman, B., and Levesque, H. J. (1992). Hard and easy distributions for SAT problems. In Rosenbloom, P. and Szolovits, P., editors, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 459–465, Menlo Park, California. AAAI Press.

Montgomery, D. C. (2001). *Design and Analysis of Experiments*. John Wiley & Sons, New York, NY, 5th edition.

More, J., Garbow, B., and Hillstrom, K. (1981). Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, 7(1):17–41.

Moret, B. M. E. (2002). Towards a discipline of experimental algorithmics. In Goldwasser, M., Johnson, D., and McGeoch, C., editors, *Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges*, DIMACS Monographs 59, pages 197–213, Providence. American Mathematical Society.

Morgan, J. and Sonquist, J. (1963). Problems in the analysis of survey data and a proposal. *Journal of the American Statistical Association*, 58:415–434.

Morrison, D. and Henkel, R., editors (1970). *The Significance Test Controversy – A Reader*. Butterworths.

Mott, N. F. and Sneddon, I. N. (1948). *Wave Mechanics and Its Application*. Oxford University Press, London.

Myers, R. and Hancock, E. (2001). Empirical modelling of genetic algorithms. *Evolutionary Computation*, 9(4):461–493.

Nagylaki, T. (1992). *Introduction to Theoretical Population Genetics*. Springer, Berlin.

Naudts, B. and Kallel, L. (2000). A comparison of predictive measures of problem difficulty in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 4(1):1–15.

Nelder, J. and Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, 7:308–313.

Nelson, B., Swann, J., Goldsman, D., and Song, W. (1998). Simple procedures for selecting the best simulated system when the number of alternatives is large. Technical report, Dept. of Industrial Engineering and Management Science, Northwestern University, Evanston, Illinois.

Neumaier, A., Shcherbina, O., Huyer, W., and Vinko, T. (2004). A comparison of complete global optimization solvers. Submitted to the special issue on Global Optimization of Math. Programming.

Newman, J. R., editor (1956). *The World of Mathematics*. Simon and Schuster, New York.

Neyman, J. (1950). *First course in probability and statistics*. Henry Holt, New York.

Noceda, J. and Wright, S. (1999). *Numerical Optimization*. Springer, Berlin.

Parkes, A. J. and Walser, J. P. (1996). Tuning local search for satisfiability testing. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI'96)*, pages 356–362.

Parsopoulos, K. and Vrahatis, M. (2002). Recent approaches to global optimization problems through particle swarm optimization. *Natural Computing*, 1(2–3):235–306.

Parsopoulos, K. E. and Vrahatis, M. N. (2004). On the computation of all global minimizers through particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):211–224.

Paulson, E. (1964). A sequential procedure for selecting the population with the largest mean from k normal populations. *Annals of Mathematical Statistics*, 35:174–180.

Pearson, E. (1955). Statistical concepts in their relation to reality. *Journal of the Royal Statistical Society*, 17:204–207.

Pichitlamken, J. and Nelson, B. L. (2001). Comparing systems via stochastic simulation: selection-of-the-best procedures for optimization via simulation. In *Proceedings of the 33nd conference on Winter simulation*, pages 401–407. IEEE Computer Society.

Pichitlamken, J., Nelson, B. L., and Hong, L. J. (2003). A sequential procedure for neighborhood selection-of-the-best in optimization via simulation. Working Paper, Department of Industrial Engineering and Management Sciences, Northwestern University (under second review for Naval Research Logistics).

Popper, K. (1959). *The Logic of Scientific Discovery.* Hutchinson, London.

Popper, K. (1979). *Objective Knowledge: An evolutionary approach.* Oxford University, Oxford.

Popper, K. (1983). *Realisim and the aim of science*. Rowman and Littlefield, Totowa, N.J.

Pukelsheim, F. (1993). *Optimal Design of Experiments.* Wiley.

Rardin, R. and Uzsoy, R. (2001). Experimental evaluation of heuristic optimization algorithms: A tutorial. *Journal of Heuristics*, 7(3):261–304.

Rechenberg, I. (1973). *Evolutionsstrategie. Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* problemata. frommann-holzboog.

Reeves, C. and Yamada, T. (1998). Genetic algorithms, path relinking and the flowshop sequencing problem. *Evolutionary Computation journal (MIT press)*, 6(1):230–234.

Rosenbrock, H. (1960). An automatic method for finding the greatest or least value of a function. *Computer Journal*, 3:175–184.

Roth, A. (1978). A new procedure for selecting a subset containing the best normal population. *Journal American Statistical Association*, 73:613–617.

Rubin, H. (1971). Occam's razor needs new blades. In Godambe, V. and Sprott, D., editors, *Foundations of Statistical Inference*, pages 372–374. Holt, Rinehart and Winston, Toronto.

Rubinstein, A. (1998). *Modeling Bounded Rationality.* MIT Press, Cambridge, MA.

Rudolph, G. (1997). *Convergence Properties of Evolutionary Algorithms.* Verlag Dr. Kovač, Hamburg.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435.

Sanders, P. (2004). Announcement of the "algorithm engineering for fundamental data structures and algorithms" talk during the summer school on experimental algorithmics, 5-7 july, 2004, kursuscentret rungstedgaard, rungsted kyst, denmark. http://www.diku.dk/forskning/performance-engineering/Sommerskole/scientific-program.html.

Sano, Y. and Kita, H. (2000). Optimization of Noisy Fitness Functions by Means of Genetic Algorithms using History of Search. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature (PPSN VI)*, volume 1917 of *LNCS*, pages 571–580, Berlin. Springer.

Santner, T., Williams, B., and Notz, W. (2003). *The Design and Analysis of Computer Experiments.* Springer, Berlin.

Santner, T. J. (1976). A two-stage procedure for selection of δ^*-optimal means in the normal case. *Communications in Statistics - Theory and Methods*, A5:283–292.

Satterthwaite, F. E. (1959a). Random balance experimentation. *Technometrics*, 1:111–137.

Satterthwaite, F. E. (1959b). REVOP or random evolutionary operation. Technical Report Report 10-10-59, Merrimack College.

Schaffer, J. D., Caruana, R. A., Eshelman, L., and Das, R. (1989). A study of control parameters affecting online performance of genetic algorithms for function optimization. In Schaffer, J. D., editor, *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, CA. Morgan Kaufman.

Schmidt, J. W. (1986). Introduction to systems analysis, modeling and simulation. In Wilson, J., Henriksen, J., and Roberts, S., editors, *Proceedings of the 1986 Winter Simulation Conference*, pages 5–16.

Schneier, B. (1996). *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* Wiley, New York.

Schonlau, M. (1997). *Computer experiments and global optimization.* PhD thesis, University of Waterloo, Ontario.

Schwefel, H.-P. (1975). *Evolutionsstrategie und numerische Optimierung.* Dr.-Ing. Dissertation, Technische Universität Berlin, Fachbereich Verfahrenstechnik.

Schwefel, H.-P. (1977). *Numerische Optimierung von Computer–Modellen mittels der Evolutionsstrategie*, volume 26 of *Interdisciplinary Systems Research.* Birkhäuser, Basel.

Schwefel, H.-P. (1979). Direct search for optimal parameters within simulation models. In Conine, R. D., Katz, E. D., and Melde, J. E., editors, *Proc. Twelfth Annual Simulation Symp., Tampa FL*, pages 91–102, Long Beach CA. IEEE Computer Society.

Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models*. Wiley, Chichester.

Schwefel, H.-P. (1988). Evolutionary learning optimum–seeking on parallel computer architectures. In Sydow, A., Tzafestas, S. G., and Vichnevetsky, R., editors, *Systems Analysis and Simulation*, volume 1, pages 217–225. Akademie–Verlag, Berlin.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology. Wiley Interscience, New York.

Schwefel, H.-P., Rudolph, G., and Bäck, T. (1995). Contemporary evolution strategies. In Morán, F., Moreno, A., Merelo, J. J., and Chacón, P., editors, *Advances in Artificial Life – Proc. Third European Conf. Artificial Life (ECAL'95)*, pages 893–907, Berlin. Springer.

Schwefel, H.-P., Wegener, I., and Weinert, K., editors (2003). *Advances in Computational Intelligence – Theory and Practice*. Natural Computing Series. Springer, Berlin.

Selvin, H. (1970). A critique of tests of significance in survey research. In (Morrison and Henkel, 1970), pages 94–106.

Shi, Y. (2004). Particle swarm optimization. *IEEE CoNNectionS – The Newsletter of the IEEE Neural Networks Society*, 2(1):8–13.

Shi, Y. and Eberhart, R. (1999). Empirical study of particle swarm optimization. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress of Evolutionary Computation*, volume 3, pages 1945–1950.

Simon, H. (1955). A behavioral model of rational choice. *Quarterly Journal of Economics*, 69(1):99–118.

Simos, T. E. and Tsitouras, C., editors (2004). *International Conference on Numerical Analysis and Applied Mathematics 2004*, Weinheim. European Society of Computational Methods in Science and Engineering (ESCMSE), Wiley-VCH.

Simpson, T. W., Booker, A., Ghosh, D., Giunta, A. A., Koch, P., and Yang, R.-J. (2004). Approximation methods in multidisciplinary analysis and optimization: a panel discussion. *Struct Multidisc Optim*, 27:302–313.

Singer, S. and Singer, S. (2004). Efficient termination test for the nelder-mead search algorithm. In (Simos and Tsitouras, 2004), pages 348–351.

Smith, V. (1962). An experimental study of competitive market behavior. *Journal of Political Economy*, 70:111–137.

So, A. and Chan, W. (1999). *Intelligent Building Systems*. Kluwer A.P.

Spall, J. (2003). *Introduction to Stochastic Search and Optimization*. Wiley.

Stagge, P. (1998). Averaging efficiently in the presence of noise. In A.Eiben, editor, *Parallel Problem Solving from Nature, PPSN V*, pages 188–197, Berlin. Springer.

Staley, K. (2002). What experiment did we just do? Counterfactual error statistics and uncertainties about the reference class. *Philosophy of Science*, 69:279–299.

Stewart, E. C., Kavanaugh, W. P., and Brocker, D. H. (1967). Study of a global search algorithm for optimal control. In *Proceedings of the 5th International Analogue Computation Meeting, Lausanne*, pages 207–230.

Sullivan, D. W. and Wilson, J. R. (1984). Restricted subset selection for normal populations with unknown and unequal varainces. In *Proceedings of the 1984 Winter Simulation Conference*.

Sullivan, D. W. and Wilson, J. R. (1989). Restricted subset selection procedures for simulation. *Operations Research*, 61:585–592.

Suppes, P. (1969a). A comparison of the meaning and uses of models in mathematics and the empirical sciences. In (Suppes, 1969c).

Suppes, P. (1969b). Models of data. In (Suppes, 1969c), pages 24–35.

Suppes, P. (1969c). *Studies in the methodology and foundation of science*. D. Reidel.

Tarski, A. (1953). A general method in proofs of undecidability. In (Tarski et al., 1953), pages 3–35.

Tarski, A., Mostowski, A., and Robinson, R., editors (1953). *Undecidable theories*. North–Holland Publ. Co.

Therneau, T. M. and Atkinson, E. J. (1997). An introduction to recursive partitioning using the rpart routines. Technical Report 61, Department of Health Science Research, Mayo Clinic, Rochester.

Trosset, M. and Padula, A. (2000). Designing and analyzing computational experiments for global optimization. Technical Report 00-25, Department of Computational and Applied Mathematics, Rice University.

Tukey, J. (1991). The philosophy of multiple comparisons. *Statistical Science*, 6:100–116.

Van Breedam, A. (1995). Improvement heuristics for the vehicle routing problem based on simulated annealing. *European Journal of Operational Research*, 86:480–490.

van der Laan, P. (1992). Subset selection of an almost best treatment. *Biometrical J.*, 34:647–656.

Watson, J.-P., Barbulescu, L., Howe, A., and Whitley, D. (1999). Algorithm performance and problem structure for flow-shop scheduling. In *AAAI/IAAI*, pages 688–695.

Weinert, K., Mehnen, J., Michelitsch, T., Schmitt, K., and Bartz-Beielstein, T. (2004). A multiobjective approach to optimize temperature control systems of moulding tools. *Production Engineering Research and Development, Annals of the German Academic Society for Production Engineering*, XI(1):77–80.

Weisstein, E. W. (2004). "Convex function". From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/ConvexFunction.html (2 April 2004).

Welch, B. L. (1947). The generalization of student's problem when several different population variances are involved. *Biometrika*, 34:29–35.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D. (1992). Screening, predicting, and computer experiments. *Technometrics*, 34:15–25.

Whitley, D., Mathias, K., Rana, S., and Dzubera, J. (1995). Building better test functions. In Eshelman, L., editor, *Proc. of the Sixth Int. Conf. on Genetic Algorithms*, pages 239–246, San Francisco, CA. Morgan Kaufmann.

Whitley, D., Mathias, K., Rana, S., and Dzubera, J. (1996). Evaluating evolutionary algorithms. *Artificial Intelligence*, 85(1–2):245–276.

Whitley, D., Watson, J., Howe, A., and Barbulescu, L. (2002). Testing, evaluation and performance of optimization and learning systems. Technical report, The GENITOR Research Group in Genetic Algorithms and Evolutionary Computation, Colorado State University.

Wineberg, M. and Christensen, S. (2004). An Introduction to Statistics for EC Experimental Analysis. CEC tutorial slides. http://ls11-www.cs.uni-dortmund.de/people/tom/public_html/experiment04.html.

Wolpert, D. and Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

Zoubir, A. M. and Boashash, B. (1998). The bootstrap and its application in signal processing. *Signal Processing Magazine, IEEE*, 15(1):56–67.

Nomenclature

Roman symbols

#	number sign, page 95
$\mathbf{1}$	vector of ones, page 94
$\overline{d}$	difference between sample means, page 22
$\overline{y}$	sample mean, page 12
$\overline{y}(v)$	mean of the node v, page 49
$\tilde{f}$	obtained best objective function value, page 94
$\tilde{x}$	position of the obtained best objective function value, page 94
$\tilde{y}$	measured response, page 74
A^T	transpose of matrix A
$B(x_0, \epsilon)$	ϵ-environment of x_0, page 62
c_i	passenger waiting bit, page 63
d	difference vector of two random samples, page 39
e	experimental outcome, page 24
$E(X)$	expectation of the random variable X, page 44
$ET(\mathcal{Y})$	experimental testing model, page 20
f	objective function
$F(A)$	size of the region A, page 140
f^*	known best objective function value, page 94
g	generation, page 85
k	number of design variables (factors), page 44

l leaf of a tree, page 49

$M(\mu)$ probability model, page 19

n sample size, page 12

n_b number of bootstrap samples, page 39

n_L the number of leaves in regression tree, page 50

n_v number of cases in node v, page 49

P probability distribution, page 19

p passenger arrival rate, page 63

$Pr(A)$ probability of event A

r sample size for reevaluation, page 126

$R(T)$ mean squared error for the tree, page 49

$R(v)$ squared error of a node, page 49

$R_{\mathrm{CV}}(T)$ cross-validation estimate for the prediction error, page 50

$R_{c_p}(T)$ cost-complexity measure, page 50

RU testing rule, page 20

S^2 sample variance, page 37

s_ν^2 estimate of σ_ϵ^2 based on ν degrees of freedom, page 129

S_d sample standard deviation of the differences, page 38

s_i server present bit, page 63

S_p^2 pooled variance, page 37

s_R estimate of the standard error of the cross-validation estimate of the prediction error, page 50

s_r success rate, page 83

$s_{1/5}$ step size update rule, page 85

SS_{E} sum of squares due to error, page 43

SS_{TREAT} sum of squares due to the treatments, page 43

SS_{T} total corrected sum of squares, page 43

T test statistic, page 20

T_L set of all leaves of a tree T, page 49

$t_{\alpha,n}$ the upper α percentage point of the t-distribution with n degrees of freedom, page 37

$U[0,1]$ uniformly distributed r.v. from $[0,1]$

v node of a tree, page 49

v_L left subtree with root node v, page 50

v_R rightsubtree with root node v, page 50

X regression matrix, page 44

$x(t)$ state of the system at time t, page 64

$X^{(0)}$ set of search points at generation 0, page 77

$x^{(0)}$ starting point, page 59

x^* minimizer (global or local), page 58

x_l lower initialization bound

x_u upper initialization bound

x^*_{ap} apparent global optimizer, page 61

x_{border} border for successful solutions, page 94

x_{effect} effects of a subset, page 53

Y random variable, page 19

y' gradient of y

$Z(\cdot)$ random process, page 52

$Z^{(\alpha)}_{k,\rho}$ upper-α equicoordinate critical point, page 37

z_α upper α percentage point of the normal distribution, page 22

$\mathcal{C}$ scientific claim, page 19

$\mathcal{D}$ experimental design, page 71

$\mathcal{D}_A$ algorithm design, page 70

$\mathcal{D}_P$ problem design, page 70

$\mathcal{F}$ regression model, page 52

$\mathcal{R}$ correlation model, page 52

$\mathcal{O}$ observation, page 20

$\mathcal{Y}$ sample space, page 20

i.i.d. independent and identically distributed r.v. random variable

Acronyms

ANOVA analysis of variance, page 43

CART classification and regression trees, page 49

CCD central composite designs

CI computational intelligence, page 62

CONT continuous update rule, page 85

CR critical region, page 20

CR critical region, page 37

CRN common random numbers, page 17

CS correct selection, page 127

DACE Design and analysis of computer experiments, page 5

DETEQ deterministically determined starting vectors, page 77

DETMOD deterministically modified starting vectors, page 77

DOE design of experiments, page 41

EA evolutionary algorithm, page 1

EC evolutionary computation, page 2

ESGC elevator supervisory group control, page 62

EXH resources exhausted, page 78

EXPIMP expected improvement heuristic, page 76

FSOL problem was solved (function values), page 78

INTV interval update rule, page 83

LHD Latin hypercube design, page 74

LHS Latin hypercube sampling, page 74

MBST mean best function value, page 95

MSE mean squared error of the predictor, page 53

NFL no free lunch theorem, page 59

NN neural network, page 62

NP Non-deterministic polynomial, page 4

NPT Neyman-Pearson theory of testing, page 20

NPT* Mayo's extension of the Neyman-Pearson theory of testing, page 24

NUNIRND non-uniform random starts, page 78

P Polynomial, page 4

PRATE progress rate, page 98

PSO particle swarm optimization, page 12

RLD run length distribution, page 95

RSM Response surface methodology, page 102

SAT propositional satisfiability problem, page 58

SCR success ratio, page 94

SPO sequential parameter optimization, page 107

SQ severity question, page 24

SR severity requirement, page 24

STAL algorithm stalled, page 78

TA threshold acceptance, page 130

TR threshold rejection, page 130

TS threshold selection, page 130

TSP traveling salesperson problem, page 4

UNIRND uniform random starts, page 78

VRT variance-reduction techniques, page 17

XSOL problem was solved, page 78

Greek symbols

$\alpha_{\overline{d}}(\delta)$ observed significance level (rejection), page 26

$\beta_{\overline{d}}(\delta)$ observed significance level (acceptance), page 28

δ difference between two population means, page 21

δ^* the smallest difference worth detecting, page 127

δ_{un} the largest scientifically unimportant value in excess of δ_0, page 26

ϵ machine precision, page 94

μ mean of Y, page 19

$\mu_{[i]}$ i-th ordered mean, page 127

Ω parameter space, page 20

π policy, page 63

π^* optimal policy, page 64

ρ correlation, page 37

σ standard deviation, page 22

$\sigma_{\overline{d}}$ standard error, page 22

τ threshold, page 130

φ progress rate, page 98

Index