JYVÄSKYLÄ STUDIES IN COMPUTING 43

Heikki Maaranen

On Heuristic Hybrid Methods and Structured Point Sets in Global Continuous Optimization

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston vanhassa juhlasalissa (S212)
joulukuun 4. päivänä 2004 kello 12.

Academic dissertation to be publicly discussed, by permission of
the the Faculty of Information Technology of the University of Jyväskylä,
in Auditorium S212, on December 4, 2004 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2004

On Heuristic Hybrid Methods and Structured Point Sets in Global Continuous Optimization

JYVÄSKYLÄ STUDIES IN COMPUTING 43

Heikki Maaranen

On Heuristic Hybrid Methods and Structured Point Sets in Global Continuous Optimization

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2004

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN 9513919943
ISBN 951-39-1994-3 (PDF)

ISBN 951-39-1904-8 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä
and ER-Paino Ky, Lievestuore 2004

ABSTRACT

Maaranen, Heikki
On Heuristic Hybrid Methods and Structured Point Sets in Global Continuous Optimization
Jyväskylä: University of Jyväskylä, 2004, 42 pages (+ included articles)
(Jyväskylä Studies in Computing,
ISSN 1456–5390;43)
ISBN 951-39-1994-3
Finnish summary
Diss.

In this work, we concentrate on improving the performance of global methods for continuous optimization via hybridization and the use of structured point sets. Optimization is an important part of solving real-life problems. The problem solving process involves modeling, simulation and optimization of the simulated model, after which the results can be applied into practice, for example, in product manufacturing. Many of the real-life problems can be formulated as global continuous optimization problems. Efficient global optimization methods are needed because realistic mathematical models are often very complex with nonconvex objective functions.

Hybridization is widely recognized to be one of the most attractive areas of method development. By hybridization we mean a combination of different methods or elements. Through hybridization, it is possible to form new methods that posses the strengths, but not the weaknesses of the original elements. Here, we construct new hybrid methods based on popular metaheuristics. We combine a simulated annealing with the proximal bundle method and a real-coded genetic algorithm with the Nelder-Mead simplex method and are able to improve both the efficiency and the reliability of the original algorithms. In addition, we form interdisciplinary hybrids by using structured point sets such as quasi random sequences and spatial point processes in initial populations of a real-coded genetic algorithm. We study the properties of the point generators and test what effects the different initial populations have on the objective function value. We also point out some difficulties in method comparison. We show that the change of test problems or other test settings may strongly affect the outcome of the comparison.

The efficiency of all the methods developed is evaluated through numerical experiments. In general, we show that hybridization, in its different forms, may be a very useful tool for improving the performance of existing methods.

Keywords: Operations research, global optimization, nonlinear optimization, continuous variables, hybridization, metaheuristics, simulated annealing, real-coded genetic algorithms, structured point sets, quasi random numbers, simple sequential inhibition (SSI) process, nonaligned systematic sampling, method comparison, test problems.

Author Heikki Maaranen

Department of Mathematical Information Technology
University of Jyväskylä
Finland

Supervisors Professor Kaisa Miettinen

Helsinki School of Economics
Finland

and

Department of Mathematical Information Technology
University of Jyväskylä
Finland

Doctor Marko M. Mäkelä

Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers Doctor Montaz Ali

School of Computational and Applied Mathematics
Witwatersrand University
South Africa

Professor Marek Rudnicki

Faculty of Technical Physics, Applied Mathematics and Computer Science
Institute of Computer Science
Technical University of Lodz
Poland

Opponent Professor Colin Reeves

School of Mathematical and Information Sciences
Coventry University
U.K.

ACKNOWLEDGEMENTS

I thank my Lord and Savior Jesus Christ for accepting me as I am and for being there for me at all times. I also wish to express my sincere gratitude to my advisers, Prof. Kaisa Miettinen and Dr. Marko M. Mäkelä, who have coauthored in almost all the papers and whose supervision has significantly improved the quality of this thesis. They have helped me to refine the papers in both scientific and linguistic aspects. They have also passed their professional expertise, knowledge and experiences.

I am grateful to Dr. Montaz Ali and Prof. Marek Rudnicki for reviewing the manuscript and giving valuable feedback. I also wish to thank Professor Antti Penttinen and Dr. Salme Kärkkäinen for their professional advise in statistical analysis and Dr. Erkki Heikkola and Prof. Tommi Kärkkäinen for their advise concerning the nature of neural networks. I also thank the many who have provided their codes for my use: Dr. Jari Toivanen for the real-coded genetic algorithm, Prof. Michael Mascagni and Yaohang Li for the Niederreiter quasi random sequence generator, Prof. Ilya Sobol' for the Sobol' generator, Dr. Erkki Heikkola and Prof. Tommi Kärkkäinen for the MLP neural network code.

This work was financed by the Academy of Finland (grants number 71822 and 102020 managed by Prof. Kaisa Miettinen). In addition, the work was supported by Jenny and Antti Wihuri Foundation, Finnish Cultural Foundation, Ellen and Artturi Nyyssönen Foundation and Jyväskylä Graduate School in Computing and Mathematical Sciences (COMAS at the University of Jyväskylä).

Last but not least, I thank my family and friends for their love and encouragement during the process. It has meant a lot to me.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

LIST OF INCLUDED ARTICLES

[A] K. Miettinen, M. M. Mäkelä and H. Maaranen, Efficient Hybrid Methods for Global Continuous Optimization Based on Simulated Annealing, Computers & Operations Research, to appear.

[B] H. Maaranen, K. Miettinen and M. M. Mäkelä, Training MLP Network Using a Genetic Algorithm as a Global Optimizer. In Resende, M. G. C. and de Sousa, J. P. (eds.) Metaheuristics: Computer Decision-Making, pages 421–448, Kluwer Academic Publishers, 2003.

[C] H. Maaranen, K. Miettinen and M. M. Mäkelä, Quasi-Random Initial Population for Genetic Algorithms, Computers and Mathematics with Applications, 47(12):1885–1895, 2004.

[D] H. Maaranen, K. Miettinen and A. Penttinen, On Initial Populations of a Genetic Algorithm for Continuous Optimization Problems, Submitted to Journal of Global Optimization.

[E] H. Maaranen, Optimization Method Selection Problem: A Practical Study Concerning Four Pitfalls, Submitted to European Journal of Operational Research.

1 INTRODUCTION

This thesis is devoted to improving efficiency of global optimization methods via hybridization and the use of structured point sets. We concentrate on nonlinear optimization problems with continuous variables. Optimization is an important part of solving real-life problems. A numerical problem solving process involves modeling, simulation and optimization of a simulated model, after which the results can be applied into practice, for example, in product manufacturing.

In the modeling and simulation phase, first a mathematical model is formed that sufficiently accurately corresponds to a physical phenomenon, process or system under investigation, and then a numerical model, which can be used in computer simulations, is formed. The models may incorporate anything from atomic level properties of materials, such as structural, thermodynamic and kinetic properties [11], to macroscopic processes, such as casting of steel [20] or business processes [37]. For simplicity, from here on, we call all objects of modeling and simulation jointly as processes.

The cost effectiveness and usability have made modeling and simulation prevalent in different areas of research. Currently, popular objects for modeling and simulation in industry are, for example, development of virtual environments and electrical components. In [12], the areas of application for simulation are divided into two main areas, manufacturing systems and public systems. Other topics include, for example, transportation, construction, restaurant and entertainment systems, business process reengineering, food processing and computer system performance [12].

The advantages of successful modeling and simulation are many [87]. For example, the process can be explored without disrupting ongoing operations, and hypotheses about how and why certain phenomena occur can be tested. New designs can also be tested without committing resources for their acquisition. For example, preliminary tests can be done in simulation and, therefore, building replica is not required until on a later stage. Hence, simulation helps detecting mistakes early in the design cycle, which lowers the repair costs dramatically. In addition, time can be compressed or expanded in order to speed up or slow down the process

under investigation. Furthermore, insight can be obtained about the interaction and importance of the variables, and "what if" questions can be answered.

However, simulation has also disadvantages [87]. For example, the benefit of the simulation is strongly related to the quality of the model. If the model is inaccurate, then simulation results may be useless. In addition, model building requires special training and simulation results may be difficult to interpret. Furthermore, modeling, simulation and analysis of the results can be time consuming and expensive, and sometimes simulated models are used, when analytic solution is possible or even preferable.

Often, these disadvantages can be overcome by the following means [87]. For example, simulation software packages are helpful for model building and they are already available and actively developed. The simulation packages often also include tools for result analysis, and, thus, the time required for the whole simulation process decreases rapidly. This is due to the increasing computational capacity of the computers and also to the development of more comprehensive and easier to use simulation packages. Finally, analytic solutions should be used, when they are preferable. However, many complex systems cannot be thoroughly analyzed using the analytical solutions.

If the simulated model is good then it accurately enough corresponds to the real process. However, when quality of the solutions plays an important role, finding a solution through simulation is not always enough. Then optimization can be used to find feasible and improved solutions for the simulated model and, hence, for the real process. This is related to the product development and repair. By a solution, we now mean the whole process, not just the end product, because it is often important to take into consideration also, for example, the energy and raw material consumption and environmental issues.

Without loss of generality, we may consider minimization problems of the form

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & LB_i \leq x_i \leq UB_i \quad \text{for all } i = 1, ..., n, \end{array} \tag{1.1}$$

where the objective function $f : \mathbf{R}^n \to \mathbf{R}$ is nonlinear, $\mathbf{x} = (x_1, ..., x_n)$ is the vector including the variables, n is the number of variables, and the feasible region $S \subset \mathbf{R}^n$ is here defined by vectors LB and $UB \in \mathbf{R}^n$ containing the lower and upper bounds for each variable, respectively.

Traditional (local) optimization methods solve convex problems, that is, problems where both the feasible region and the objective function are convex. However, many times, mathematical and physical theory behind real-life processes is very complex and realistic modeling of continuous real-life problems often results in nonconvex minimization problems, which may have several local minima. They are hard to solve and, therefore, require efficient global optimization methods. The increased computational capacity of computers have made it possible to solve optimization problems with more and more accurate simulated models. The availability of new resources has aroused a lot of interest in the last few decades leading to increasingly growing number of research papers including [8, 9, 22, 23, 47, 86, 96] and books [36, 52, 85] published on global optimization containing new methods

for solving problems with continuous variables.

In this thesis, we concentrate on improving the efficiency of global continuous optimization methods by using hybridization and structured point sets. In hybridization, different methods are combined to form new methods. The aim in hybridization is to preserve the advantages and tackle the disadvantages of the original methods. Hybridization is widely recognized as one of the most attractive new trends in global optimization in general [54, 88, 90]. The use of structured point sets [28, 39, 49], on its behalf, has provided promising results in various fields of science, such as, simulation [56, 62], numerical integration [17, 81, 89, 103, 115], and random search methods [63, 81, 98, 104]. However, the idea has not yet been widely exploited in optimization. Before going further into these two focus areas, let us first give a short overview of the field of continuous optimization in general.

1.1 Local Optimization

Traditional methods in continuous minimization use gradients to determine the most attractive search directions for finding a local minimum of the objective function $f : \mathbf{R}^n \to \mathbf{R}$. A solution $\mathbf{x}^*_{loc}$ is a *local minimum*, if there exists some $\delta > 0$ such that $f(\mathbf{x}^*_{loc}) \leq f(\mathbf{x})$ for all $\mathbf{x} \in S$ satisfying $\|\mathbf{x} - \mathbf{x}^*_{loc}\| < \delta$.

Next, we define a concept of decent direction that is closely related to minimization. A direction $\mathbf{d} \in \mathbf{R}^n$ is said to be a *decent direction* at $\mathbf{x}$ if $\nabla f(\mathbf{x})^{\mathrm{T}}\mathbf{d} < 0$, since this implies that there exists $t > 0$, such that $f(\mathbf{x} + s\mathbf{d}) < f(\mathbf{x})$ for all $s \in (0, t]$. For direct search methods, which do not use gradient information, a decent direction is usually defined experimentally: if there exists a sample point $\mathbf{x}'$ so that $f(\mathbf{x}') < f(\mathbf{x})$, then the a move from the point $\mathbf{x}$ to $\mathbf{x}'$ is said to be a move to a decent direction. Collectively, the methods following only descent directions are called *local search methods* or *local (optimization) methods* and problems where it is sufficient to find a local minimum are called *local optimization* (or *minimization*) *problems*.

Probably the best-known gradient-based method is the *steepest descent method* [55], that repeatedly moves to a steepest decent direction until the gradient ∇f becomes zero (within some tolerance $\varepsilon > 0$). To be more exact, the new point $\mathbf{x}'$ is obtained from the current point $\mathbf{x}$ by the formula

$$\mathbf{x}' = \mathbf{x} + s\mathbf{d},$$

where $\mathbf{d}$ is the negative gradient direction $-\nabla f(\mathbf{x})$ and the step size s is chosen by a line search [14].

Gradient-based methods are often powerful tools for problems that are differentiable and convex. Some of the disadvantages of the traditional gradient-based methods become evident when the above mentioned very restrictive conditions are not met.

For continuous nondifferentiable problems, several methods have been developed that do not require gradients. They approximate the negative gradient direction either by using subgradients like the bundle methods [58, 69], or by using only

objective function values like the direct search methods. Well-known examples of direct search methods are Nelder-Mead simplex method [79] and the Hooke and Jeeves direct search method [51]. The methods that only approximate the negative gradient direction can be applied to a wider range of continuous minimization problems than the gradient-based methods, because they do not require the objective function to be differentiable. However, all the traditional direct search methods and methods that approximate gradients as well as gradient-based methods terminate the search after finding the first local minimum.

1.2 Global Optimization

Many times it is desirable to search for the best possible local minimum, which is called a global minimum. Mathematically we define that a solution $\mathbf{x}^*$ is a *global minimum* if $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in S$. Here, we consider problems, where it is sufficient to find one global minimum and we call the problem a *global minimization problem*. The methods that are designed to tackle global minimization problems are called *global search methods* or *global (optimization) methods*.

Next, we give a definition and formulate a theorem which are both closely related to global minimization. We define the *basin of attraction* of a local minimum $\mathbf{x}^*_{loc}$ as the largest set of points such that for any starting point from that set the infinitely small step steepest descent algorithm will converge to $\mathbf{x}^*_{loc}$ [112]. We also formulate the following theorem

Theorem 1 Let S be a nonempty convex set in $\mathbf{R}^n$ and $f : S \to \mathbf{R}$ and consider the problem to minimize $f(\mathbf{x})$ subject to $\mathbf{x} \in S$. Suppose that $x^* \in S$ is a local optimum to the problem. If f is convex, then $\mathbf{x}^*$ is a global optimum.

For a proof of the theorem, see, for example, [14]. The condition of Theorem 1 that f is convex can be relaxed, for example, to strict quasiconvexity. Furthermore, in our case, feasible regions are always nonempty and convex, since we defined them by using lower and upper bounds for each variable. Hence, if the minimization problem is of the form (1.1), then, according to Theorem 1, the convexity of the objective function f is sufficient to guarantee that a local optimum is also a global optimum, and, thus, local optimization methods can be used.

In addition to convex problems, local search methods usually work well for *multimodal* problems, that is, problems with more than one local minimum, if a good initial guess is available or if the number of local minima is small and the basin of attraction of a global minimum is large. In these cases, the probability that the starting point is located in the basin of attraction of a global minimum is quite high. However, many real-life problems have several local minima and no good initial guesses are necessarily available. Then local search methods generally fail to find a global minimum. Indeed, practical global continuous optimization problems are in general very difficult to solve to the optimality.

Next, we further motivate the use of global optimization methods. We note that even though the conditions in Theorem 1 may be relaxed, they are still quite restrictive [10, 14]. Hence, in a general case, we cannot assume that the objective function would have only one local minimum and one basin of attraction. In addition, it is, in practice, often hard to prove the convexity or strict quasiconvexity, for example, if the problem has a black box type of objective function. Hence, global optimization methods are needed.

Global optimization methods either *operate globally* on several solutions at the same time or they *operate locally* and include a mechanism that enables them to escape the basin of attraction of a local minimum in order to search for better local minima and eventually a global minimum. Methods that guarantee to find a global minimum (within some positive tolerance) using deterministic rules are called *exact (global) methods.* Examples of exact methods are covering methods [112] (e.g., branch and bound type of methods) that iteratively detect and exclude subareas not containing a global optimum. The conditions for using exact methods are quite restrictive and, therefore, the exact methods can usually be applied only for some specific problems. In addition, for complex problems the exact methods are very time consuming. Due to these two reasons, heuristic methods have been developed for solving global minimization problems. *Heuristic methods* use rules of thumb when searching for a global optimum, but, in practice, they cannot guarantee to find it. Heuristic methods are often stochastic, which means that they involve a certain degree of randomness. For many stochastic methods there exist theoretical convergence proofs. For example, stochastic convergence means that the algorithm converges to a global minimum with probability 1, that is, the probability for convergence approaches one as the running time goes to infinity.

A good reference for both exact and heuristic global search methods is [112]. Heuristic methods include, but are not restricted to, clustering methods, level set methods, generalized descent methods and random multistart (restart) methods. More recently, methods called metaheuristics have been developed.

1.3 Metaheuristics

Metaheuristics [24, 42, 83, 94, 118] are heuristic methods that contain some sort of a metastrategy that guides the heuristic search towards a global optimum. The concept of a metastrategy is not well-defined and, therefore, also the difference between heuristic methods and metaheuristics is sometimes unclear. The list of metaheuristics often includes, but is not limited to, genetic algorithms [26, 46, 50, 73, 75, 74], simulated annealing [6, 15, 53, 57, 84, 100, 116, 119], tabu search [2, 13, 21, 25, 40, 43, 99], scatter search [40, 41, 42, 44, 59, 61], differential evolution [24, 83, 91, 107, 108], ant colony and other particle swarm optimization [16, 30, 31, 32, 33, 34, 114, 117, 121], memetic algorithms [24, 27, 72, 78] and controlled random searches [7, 92]. The main emphasis of the above listed references is on methods capable of solving problems with continuous variables. However, many of the metaheuristics have their roots in discrete optimization and, therefore, the lists

also include some established references to those methods. As the names suggest, many of the metaheuristics imitate some natural phenomenon such as annealing of steal or glass, natural genetic and cultural evolution, bird flocking and behavior of ant colonies.

Metaheuristics can be divided into single solution -based and population-based methods. In *single solution -based* methods, like simulated annealing and tabu search, the neighborhood of one solution is considered at a time. These methods essentially move from one local minimum to another and they use some algorithm-specific escape mechanisms in order to avoid stagnating to a local minimum. For example, in simulated annealing, non-descent moves may be selected and, in tabu search, the search is directed to attractive and less-explored areas by declaring some sets of solutions forbidden or "tabu." In addition, the balance between global and local operations in single solution -based methods can be adjusted by adapting the size of the neighborhood considered. There are also population based variants of simulated annealing [9] and tabu search. The population-based tabu search method is called a scatter search, which was developed in parallel with tabu search, and includes a similar ideology.

In *population-based methods*, such as genetic algorithms, differential evolution, scatter search, particle swarm optimization and memetic algorithms, several solutions are considered at the same time. These methods operate globally by scanning the whole feasible region to some degree before concentrating on the promising areas. The initial populations of all the basic versions of the population-based methods are uniformly distributed over the feasible region, but operations how the population is evolved are different. In genetic algorithms, the population is evolved by using genetic operators like selection, crossover and mutation, whereas differential evolution evolves the population by adding a weighted difference of two solutions in a population to a third solution. In particle swarm optimization, the search is guided by moving the solutions towards the all-time best and current best solutions according to prescribed rules. Memetic algorithms, on the other hand, combine local search heuristics to the operators of genetic algorithms. Memetic algorithms are, therefore, usually called hybrid genetic algorithms.

1.4 Hybrid methods

In hybridization, elements from different origins are combined in a way that they form a new optimization method. The aim of hybridization is to combine the elements so that they complement each other in the best possible way. The *elements* do not necessarily have to be complete optimization methods, but they may be any components that can be embedded in methods. In hybridization, it is desirable that the new hybrid method possesses the strengths, but not the weaknesses, of the original elements. One common way to form a hybrid is to combine a global method and a local method. Then, the aim is to combine the efficiency and accuracy of the local method to the reliability of the global method as illustrated in Figure 1. By *efficiency* and *accuracy* we mean the total computing time used by the algorithm

and how close the final solution is to the nearest local minimum, respectively. On the other hand, by *reliability* we mean, how well the algorithm finds the global minimum. Hence, if we manage to form a hybrid that is efficient, accurate and reliable, then it converges fast to the proximity of a global minimum.

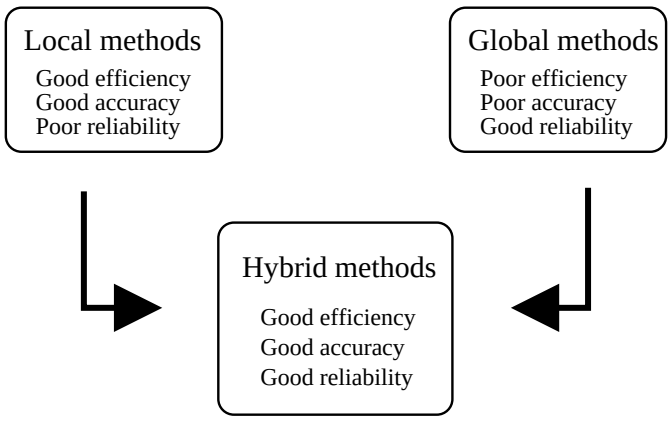

FIGURE 1: Hybridizing a local method and a global method.

Metaheuristic methods and different hybridizations of them are more commonly used for discrete than for continuous problems. However, there is an increasing number of continuous hybrid metaheuristics available, as well, and we do not intend to give an exhaustive list of them here. Many of the above-mentioned more or less established benchmark metaheuristics are not "pure", but are, in fact, hybrid methods. Some other recent hybrid metaheuristics for continuous optimization are, for example, [9, 48, 124, 125] for simulated annealing, [22, 47, 71, 106] for genetic algorithms, [23, 96, 113, 126] for tabu search and [117] for particle swarm optimization.

Hybrid methods can be characterized using different taxonomies [90, 110, 123]. The main part of the taxonomy in [90] is illustrated in Figure 2, where sequential and parallel hybrid methods form the two main branches. Note that the hybridizations considered in [90, 110, 123] are limited to combinations of different optimization methods, not elements as we have defined here.

In the part of the taxonomy illustrated in Figure 2, the division is made based on the order in which the combined methods are executed. In sequential hybrids, the previous method is executed from start to finish before the next one is started, whereas in parallel hybrid methods, the next method starts before the previous one ends. The parallel methods have synchronous and asynchronous methods as subbranches. In synchronous methods, the execution order is predefined, whereas for asynchronous methods, the execution order may be different on each iteration. For a more detailed presentation of the taxonomy, see [90, 110].

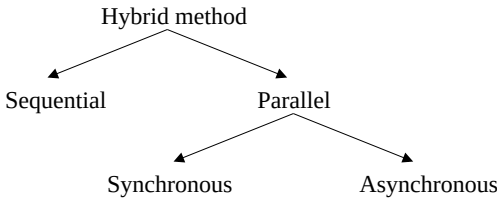

FIGURE 2: Taxonomy of hybrid methods.

1.5 New hybrid methods

Henceforth, we concentrate on the work done in papers [A]-[E]. In [A], we presented new hybrid methods that combine a simulated annealing [45] and the proximal bundle method [58, 69]. Simulated annealing is a global search method and the proximal bundle method is a local search method that uses subgradients and whose speed of convergence is comparable to the gradient-based methods [64]. The aim in this hybridization was to combine the reliability of simulated annealing to the efficiency and accuracy of the proximal bundle method. The total number of simulated annealing based global search methods that we developed and tested was 32, and four hybrids were presented. In the literature, simulated annealing has been a popular building block for hybrids. In [84] alone, 13 optimization methods based on simulated annealing were presented and 8 of them were hybrids.

In [A] the new hybrid methods were called biased proximal bundle, hybrid A, hybrid B and hybrid C, and, although they are all parallel synchronous hybrids, they were constructed in different ways. Figure 3 shows the main structure of the hybrids, where SA stands for simulated annealing, PB for the proximal bundle method and $\mathrm{PB_a}$ for proximal bundle method with variation of accuracy.

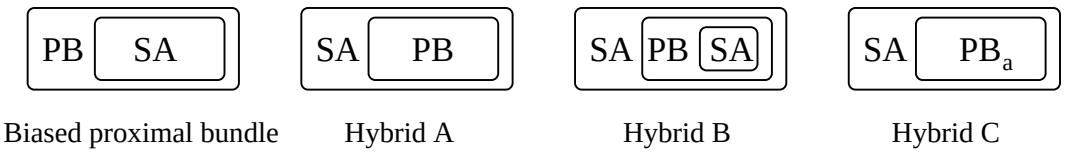

FIGURE 3: Structure of the hybrids.

In our hybrids proposed in [A], we used the simulated annealing described in [45]. As mentioned earlier, simulated annealing considers only one solution at a time. A new trial solution is generated from the current solution by varying each component according to prescribed rules, (see [A] or [45], for further details). The trial solution $\mathbf{y}$ is accepted and becomes a current solution of the next iteration if the trial solution has a better objective function value than the current solution $\mathbf{x}$ or if a so-called Metropolis criterion is valid. In other words,

$$e^{(f(\mathbf{x})-f(\mathbf{y}))/t} > p,$$

where p is a uniformly distributed random number from $[0, 1]$. Here, the parameter $t > 0$ is a so-called temperature and it is decreased during the algorithm.

Our original idea was to embed the Metropolis criterion of simulated annealing in the proximal bundle method, so that it could escape local minima. The method was called *biased proximal bundle.* Unfortunately, the results of the numerical tests were not promising. We assume the the directions provided by the proximal bundle were "too good" and the Metropolis criterion was not strong enough a mechanism to efficiently force the search out of the basins of attraction of local minima.

In *hybrid A*, we used the proximal bundle inside simulated annealing and not vice versa as in biased proximal bundle method. Now, the proximal bundle method was applied starting from a candidate solution accepted in simulated annealing. *Hybrid B* is otherwise the same as hybrid A, but now the proximal bundle method was replaced by the biased proximal bundle method. In this way, both methods combined in hybrid B could escape a basin of a local minimum before searching the local area exhaustively. Hybrid B was in [A] called a *two-level parallel synchronized hybrid*, because it combines two hybrid methods. In *hybrid C*, the proximal bundle method is started with a relatively low accuracy and the accuracy is increased towards the end of the search. The control parameter for the increase of accuracy of proximal bundle method was defined to be proportional to the temperature of simulated annealing. This way both the methods intensify the search in parallel. Moreover, low accuracy of the local search during the early iterations decreases computational cost of hybrid C when compared to hybrid A.

Parallel synchronous hybrids based on simulated annealing are presented also, for example, in [48, 96]. Our methods in [A] differ from those in [48, 96] in some significant ways. For example, we used a different local search method and especially the hybrid B has quite a different structure.

The performances of the methods were evaluated by solving 38 test problems from the literature. The numerical experiments in [A] show that the hybridization can improve both the efficiency and the reliability of simulated annealing. However, when comparing the efficiency we should keep in mind that the hybrid methods use subgradients, which increases the computational efforts, but which does not show in the number of function evaluations. As benchmark methods we used the original simulated annealing of [45] with the parameter values recommended in [45] and [101]. In general, the parameter values used in [45] emphasize reliability at the expense of computational cost, and the converse is true for the values used in [101]. However, the hybrids A, B and C were more robust than the original algorithm emphasizing reliability and more efficient than the one emphasizing efficiency. This indicates that the hybridization was well realized.

We evaluated the efficiency by using the number of function evaluations and the reliability using success rates among other things. The test problems were solved ten times with each method and a run was considered to be successful if the difference between the final objective function value and the know optimal value was less than 0.1. The best of the hybrid methods was hybrid C. It had a 90.5% average success rate of finding a global minimum, and it used on the average 12,284 function evaluations, where the averages are taken over all the

runs and over all the test problems. The respective values for the benchmark methods were 73.9% and 58.4% for the success rate and 3,089,906 and 104,007 for the average number of function evaluations. Let us remind that the number of objective function evaluations are not quite comparable since the proximal bundle method used in hybrids computes also subgradients. However, the hybridization improved the reliability of the methods quite significantly, and often for global optimization methods reliability is considered to be more important than efficiency.

In [B], we presented another parallel synchronous hybrid method, where we again combined a global search with a local search. This time we combined a real-coded genetic algorithm with the Nelder-Mead simplex method so that the hybrid method operates directly on function values not requiring any gradient information. Consequently, this type of a hybrid can be used for a wide range of problems. We applied it for training a Multi Layer Perceptron (MLP) neural network for solving a regression problem. Genetic algorithms have been used for training neural networks also, for example, in [97], where three different types of neural networks are trained: MLPs, radial basis functions and probabilistic neural networks. Other metaheuristics, such as tabu search [66] and scatter search [60] have also been applied to train neural networks. In [70], twelve different training (optimization) methods are compared.

The initial idea in [B] was only to use a modern metaheuristic for training the MLP neural network. First, a scatter search implementation was applied but the results were discouraging. Then we applied the real-coded genetic algorithm with similar results: the final objective function values were inferior to the ones of a simple multistart method using a gradient-based local search method that was being used as a benchmark. In a simple multistart method, a local search is started from random trial points. Then, we hybridized the real-coded genetic algorithm with a local search method using gradient information, and the accuracy of the solutions improved considerable while the speed also remained better than that of the multistart method. Finally, the gradient-based local search method was replaced both in the multistart and hybrid genetic algorithm by the Nelder-Mead simplex method, so that no assumptions about the differentiability of objective functions had to be made. In [B], only the results of the final hybrid versions are reported.

The implementation of the hybrid genetic algorithm was kept simple. In the first phase of the algorithm, the Nelder-Mead simplex method was run on 10 random starting points within the specified feasible region, and the solutions were added to the initial population of the genetic algorithm. This improved the performance of the genetic algorithm considerably. However, the number of added local minima was kept small since adding too many local optima to the initial population could increase the danger of loosing diversity within the population. This might lead to a premature convergence to a local minimum far from a global minimum. In the second phase, the genetic algorithm was run from start to finish, and, in the end the, the Nelder-Mead simplex method was again applied to the best solution found by the genetic algorithm.

The performance of the hybrid genetic algorithm was evaluated by solving a

regression problem and by comparing the results to those of a simple multistart method. In a regression problem, given a noisy data set, the aim is to find a mapping that approximates the original data without noise. For the numerical experiments, we generated 20 sinus curves with normally distributed random noise. Each noisy sinus curve represents one test problem and each test problem was solved 20 times with both of the methods. Now, the task for the methods was to recover the original sinus curve based on 50 training data points for each test instance. The original sinus curve and one set of training (noisy) data is illustrated in Figure 4.

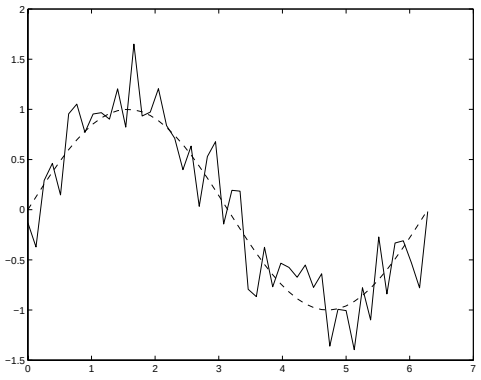

FIGURE 4: The original sinus curve and one noisy training data set.

The performance of the proposed method was validated by using numerical experiments and statistical tests. The analysis of the results showed that the hybrid method is both more efficient and more reliable than the simple multistart method. The hybrid genetic algorithm used on the average 65,000 function evaluations, where the average values were computed over all the runs and over all the test problems. This is far less than the average of 150,000 function evaluations used by the multistart method. The accuracy of the solutions of the hybrid genetic algorithm was also better than that of the multistart as can be seen in Figure 5, where 20 solution curves are plotted for the above described training data set.

Let us summarize that, in [A] and [B], the hybrid methods were formed by combining two or more optimization methods. If a hybridization is defined in a general way, as we have done, then embedding any ready-made element in an optimization method is called hybridization. If the embedded element is not an optimization method, we refer to the method as an *interdisciplinary hybrid*. Next, we consider interdisciplinary hybrid methods that use structured point sets for generating initial populations of genetic algorithms.

1.6 Structured point sets

In [B], we were able to obtain good results by changing the initial population of genetic algorithms by adding some local minima to the initial population. This

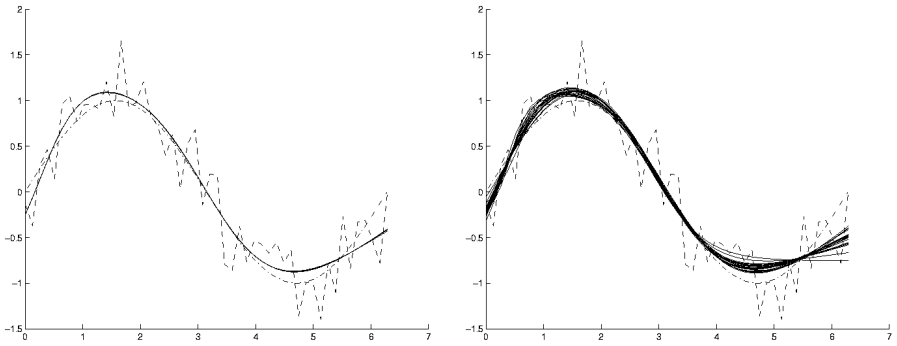

FIGURE 5: 20 solution curves obtained with hybrid genetic algorithm (on the left) and multistart method (on the right).

motivated us to study the initial populations of genetic algorithms more carefully in [68], and we continued the work in [C] and [D]. We studied the case when no information about the local minima is available to see whether we could find better initial populations than the traditional pseudo random ones. To that end, we selected some quasi random sequence generators and spatial point processes. Spatial point processes are used in statistics [28] and quasi random sequences are used in numerical integration [18, 102, 105], computer simulations [1, 65] and quasi random searches [63, 80, 104] with good success. In addition, quasi random sequences have been used in generating initial solutions for modified controlled random searches [4] and topographical multilevel single linkage [5], again with good success. Both quasi random sequence generators and the selected spatial point processes are designed to produce points that maximally avoid each other.

Very little research has been done on the generation of an initial population of genetic algorithms. Traditionally, an initial population is said to be selected randomly, which, in practice, means that pseudo random numbers [39, 49] are used. It is a well-known fact that truly random numbers cannot be generated algorithmically [103], and, in practice, pseudo random numbers are designed only to imitate random numbers. However, the question that is often left unasked is whether the initial population should be random (or imitate randomness) in the first place [93]. In [C] and [D], we approached this issue by considering different features of point generators and studying their effects on the performance of a real-coded genetic algorithm. We use the term *structured point sets* jointly for all point sets that do not imitate random points, but are designed to be very evenly distributed over the feasible region. We say that a point set has a *good uniform coverage* if the points are well spread out to cover the whole feasible region.

Our hypothesis in [C] and [D] was that points that do not form clusters and yet have some degree of randomness are well suited as initial populations of genetic algorithms. In [D], we clarified this assumption with the following academic example. We selected so-called Griewangk [38] and Katsuura [35] functions as the test problems and generated heavily clustered initial populations, where each variable

was restricted to a subinterval with only 80% of length of the variable's total range. The first clustered initial populations were generated so that the global optimum was outside the subspace and the second clustered populations so that the global optimum was inside the subspace. As a benchmark we generated initial populations with solutions spread over the whole feasible region. All the solutions in this test were generated using a pseudo random number generator. See Figure 6 for a two-dimensional illustration of the test settings, where the large cross illustrates the global minimum and the range of each variable is $[-0.1, 1)$.

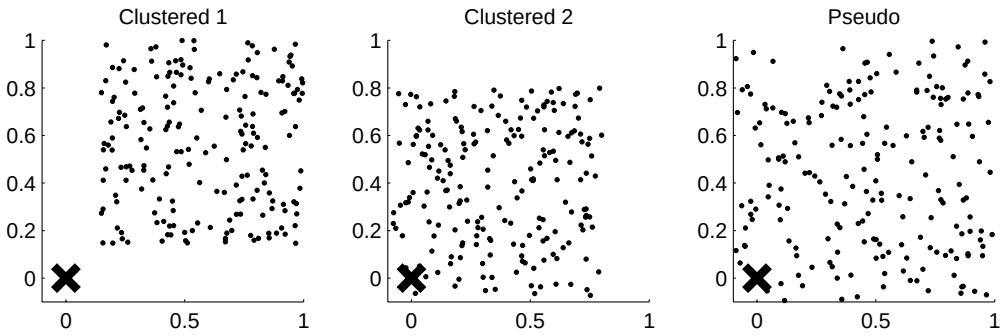

FIGURE 6: Two clustered populations and one pseudo random population spread over the whole feasible region.

Here, we consider only the 10-dimensional Katsuura function, which has the following definition

$$f(\mathbf{x}) = \prod_{i=1}^{10} \left(1 + i \sum_{m=0}^{30} \frac{\mid 2^m x_i - \lfloor 2^m x_i \rfloor \mid}{2^m} \right), \qquad -0.1 \leq x_i < 1,$$

where $\lfloor \cdot \rfloor$ is a floor function. The global minimum for the Katsuura function is at $\mathbf{x}^* = (0, ..., 0)$ with $f(\mathbf{x}^*) = 1$. Note that the feasible region of the Katsuura function has been adjusted for this experiment. For the experiments with the Griewangk function we refer to [D]. Figure 7 shows the convergence curves of the genetic algorithm with different initial populations. Each curve represents an average convergence of 100 independent runs. The effects of the different initial populations are clear even after several dozens of generations as can be seen in Figure 7. The fastest convergence was obtained, when the initial population was clustered around the global minimum. Respectively, the slowest convergence was obtained when the initial population was clustered away from the global minimum, and the evenly distributed pseudo random initial population provided convergence curve that was between the other two curves.

Based on the above described simple experiment, we may conclude that the initial population has effects on the convergence for several iterations. Hence, it is justified to study the role of initial populations of genetic algorithms. However, the above described simple experiment is only an academic example with very strong clustering. In [C] and [D], we consider more realistic ways to generate initial populations using structured point sets.

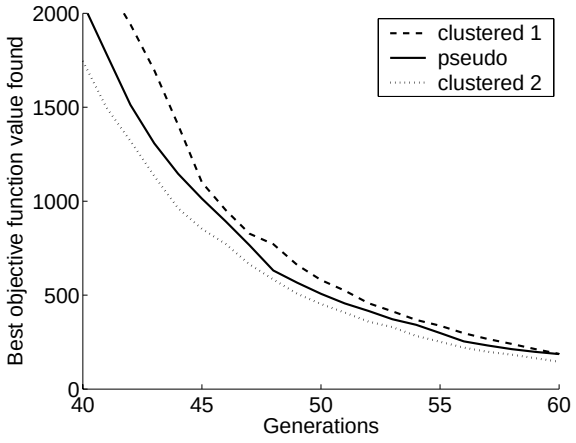

FIGURE 7: The convergence of a genetic algorithm for a Katsuura function with different initial populations.

Figure 8 shows a simple two-dimensional example of point sets generated with a pseudo random point generator, a Niederreiter quasi random sequence generator, a simple sequential inhibition (SSI) and a nonaligned systematic sampling. There is a distinct difference between the populations. The pseudo random points seem random, whereas structured point sets have more or less clear patterns and a better coverage with less clusters.

In [C], we studied quasi random initial populations. Quasi random sequences [39, 49, 81] are designed to produce points that "maximally avoid each other" [89]. As mentioned earlier, they have been successfully used, for example, in numerical integration, simulation and quasi random searches. We compared the results of the genetic algorithms using the so-called Niederreiter [81] and Sobol' [17] quasi random sequences to those of the original genetic algorithm using pseudo random sequences. The Niederreiter and the Sobol' quasi random sequence generators were selected since they had performed well in our earlier tests reported in [67].

Quasi random sequences are also called low-discrepancy or low-dispersion sequences. *Discrepancy* and *dispersion* are measures of coverage in a sense that their low values indicate good uniform coverage. In a point set, discrepancy is large if there exist clusters or large unexplored areas, and dispersion is large if there exist large unexplored areas. Hence, every low-discrepancy sequence is also a low-dispersion sequence, but not vice versa. For our purposes, we are interested in point sets that have low values for discrepancy or dispersion. For a mathematical relationship between discrepancy and dispersion, see [81].

In the comparison of the different generators, we used a test suite of 52 computationally difficult global optimization problems (see [C] for further details). We classified them into two problem sets, P1 and P2, depending on the empirical experiments on their complexity and assigned two sets of optimization parameter values, respectively. We solved all the problems a hundred times with each of the three implementations with different initial populations applying pseudo random numbers, and the Niederreiter and the Sobol' quasi random sequences. The re-

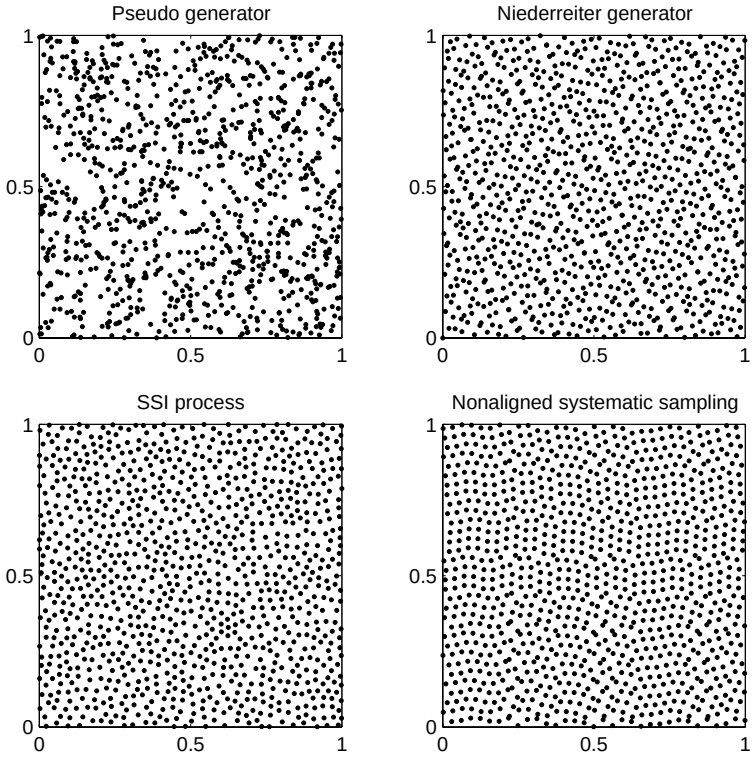

FIGURE 8: Point sets of 1024 points generated with different point generators.

sults indicated that the distribution of the initial population may have an effect on both the final objective function values and the number of generations used. In about two thirds of test problems, the genetic algorithm applying quasi random sequences improved the average final objective function values. The trade off between the solution quality and the number of function evaluations used was better than expected. The genetic algorithm using the Niederreiter quasi random initial population used even less function evaluations than the original algorithm and, while using the Sobol' sequences, the increase of function evaluations was very moderate. Despite the promising results, the effects of different initial populations were hard to interpret and further research was needed.

In [D], we continued the study of initial populations of the genetic algorithm, which was started in [C]. We added more statistical tests and more point generators. Now, we compared the original genetic algorithm using pseudo random numbers and the one applying Niederreiter quasi random sequences that was evaluated as the best quasi random genetic algorithm in [C] and in [67]. We also complemented the comparison with genetic algorithms using initial populations generated by two spatial point processes [28], which are point generators commonly used in statistics. From the various spatial point processes we selected simple sequential inhibition and nonaligned systematic sampling, which had proved promising in our preliminary tests.

First, we studied the properties of the generators. We concentrated on ana-

lyzing the uniform coverage and the genetic diversity of the points as well as the speed and the usability of the generators. By *speed* we mean how fast the points are generated and *usability* stands for both the availability and usability of the generators. The availability means, how easy it is to obtain a well-tested implementation of each generator. By a *genetic diversity* we mean the ability to reach as large part of the feasible region as possible by means of crossover. Crossover is an operator imitating genetic recombination, which forms new solutions by crossbreeding selected parent solutions in the population. We point out that all the above-mentioned properties of the point generators are related to the dimension of the problem.

We found out that the generators have different strengths and weaknesses, which was expected, because a good uniform coverage and genetic diversity of the points are somewhat conflicting properties. This is shown in Figure 9, which illustrates points and connecting lines of a rectangular grid, triangular grid, clustered points and seemingly random points without clustering. The points in Figure 9 illustrate the solutions in a population and the lines illustrate the point sets that can be reached by means of heuristic crossover [73], that is, the lines provide some idea of the genetic diversity of the points. By looking at Figure 9, it seems that the best genetic diversity is obtained using seemingly random points with no clustering. On the other hand, it is a well known fact that we obtain optimal circle packing using honeycomb-like hexagonal packing [120], that is, using triangular grid. From the packing results we immediately deduce that the maximum distance between points and, thus, good uniform coverage, is obtained using a triangular grid. Hence, genetic diversity and good uniform coverage may be somewhat conflicting criteria.

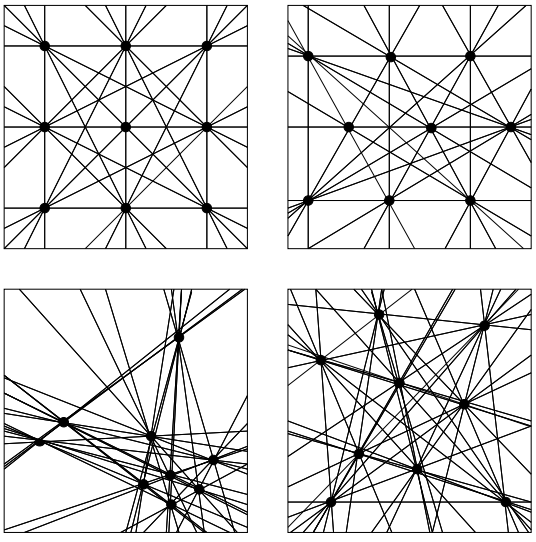

FIGURE 9: Points and connecting lines for a rectangular grid, triangular grid, clustered points and seemingly random random points with no clustering.

Table 1 shows the summary of the evaluated properties of the different point generators. On the first column of Table 1 are the property descriptions and the

following four columns list the evaluations for each generator. As before, SSI stands for simple sequential inhibition. In the property evaluation, it was suitable to call problems small, if there were 5 or less variables and large, otherwise. If a generator scored differently for small and large problems, then these scores are separated with a slash, for example, good/poor means that the generator performed well for small problems but poorly for large ones. For more details, see [D].

Properties (small/large)	Pseudo	Niederreiter	SSI	Nonaligned
Coverage	poor	moderate	good	good/poor
Genetic diversity	good	moderate	moderate	poor
Speed	good	good	poor	moder./good
Usability	good	good	moderate	good/poor

TABLE 1: Summary of properties for four generators.

With respect to the coverage properties of the different point sets, we point out that even the stuctured point sets have their shortcomings. Good coverage properties of quasi random sequences have been shown to degrade with dimension [77, 111], the points generated with SSI process tend to concentrate on edges of the feasible region [111], and nonaligned systematic sampling reduces to pseudo random sampling in higher dimensions.

After testing the properties of the generators, we used them for generating initial populations for the real-coded genetic algorithm used in [C]. In the numerical experiments, we used the same test suite as in [C], but this time classified the problems into two subclasses depending on the number of variables. In the first subclass, we put the problems with ten or less variables and the rest of the problems we put in the second subclass. The results were similar to those in [C]. They gave some indication that initial populations with good coverage may prevent premature convergence to a non-optimal solution and that good genetic diversity may fasten the convergence. However, the differences in the results were small. In fact, even though there were some differences, they were mainly not statistically significant.

Now that we have summarized the results reported in [C] and [D], let us go a little bit further using the insight obtained during the research process. By looking at Figure 7 and considering only the coverage of the initial population, we should not expect that the various initial populations would cause much difference on the average objective function values, since, even if there were some harmful clusterizations, there is also expected to be some beneficial clusterizations, and these may cancel out each other's effects. We should also not expect to see any differences if the algorithm is run long enough, since genetic algorithms are Markov chains (see, e.g., [82, 122]) and they are expected to converge to a global optimum after a sufficient number of generations. However, this sufficient number of generations may be very large and there is no practical way to ensure that the algorithm has converged. Therefore, the algorithm may in practice often have to be stopped before the global optimum is found. By looking at Figure 7 and considering only the coverage, we could expect to see some differences in the magnitudes of the

variances, if the algorithm was interrupted after a small number of generations.

Since the emphasis in papers [C] and [D] was on average objective function values, we feel obligated to complement the results reported in [C] and [D] by shortly considering variances after a small number of generations of the genetic algorithm. Here, we ignore the magnitudes of the variances and consider only the number of problem instances when the variants of genetic algorithm using structured point sets had a smaller average variance in the objective function value than the original algorithm. The check points are selected to be after 10 and 20 generations. The results are given in Table 2. No division to small and large problems is made, instead, all the 52 problems are considered at the same time. Hence, the maximum value in Table 2 would be 52.

	Niederreiter	SSI	Nonaligned
After 10 generations	21	17	21
After 20 generations	26	23	31

TABLE 2: Number of instances when the average variance was smaller than that of the original genetic algorithm.

The results in Table 2 indicate that even in the variances of the early generations there is no clear difference between the genetic algorithms using structured point sets and the original genetic algorithm. As a conclusion for the research done in [C] and [D], along with the brief complementary study here, we may say that the results obtained with different types of point generators are, in essence, the same. This means that the genetic algorithm tested is very robust with respect to changes in initial populations. Moreover, the fast, easy to use and readily available pseudo random sequence generators may continued to be used to generate initial populations to the type of genetic algorithm tested here.

Let us note, however, that we reported here the first results concerning the initial populations for population-based metaheuristics in global continuous optimization. The results cannot not be directly transferred to all problems or all types of genetic algorithms, not to mention all population-based metaheuristics. Further research is needed to find out the whether other methods are as robust as the genetic algorithm tested.

1.7 Method selection problem

While evaluating the hybrid methods proposed in [A]-[D], we noticed how hard it is to compare different methods. Even the comparison of two similar methods is demanding, as stated in [95], where two genetic algorithms were compared. When the methods are inherently different, a fair comparison becomes very challenging, for example, because it then involves separate tuning of the optimization parameter values.

In [E] we considered the method selection problem for global continuous optimization. The selection of an optimization method is based on its performance,

which is usually evaluated using measures such as reliability, efficiency and accuracy as mentioned earlier. We concentrated on the case, where one repeatedly needs to solve similar type of problems and wants to find the best method for the task. The results in [E] are also helpful in method development.

The method selection process can be described with four steps. On the first step, one selects the test problems so that they, as accurately as possible, represent the whole set of problems that will be solved with the method. On the second step, one selects the optimization methods for the comparison. If one has solved similar problems earlier, the methods used can be included as benchmark methods. One should also study what kind of methods have been used in the literature for solving that type of problems and select the most attractive methods and then solve the problems. The third step involves the analysis of the results. If none of the methods has acceptable performance, one needs to return to step two and select new methods or go to step four and try to improve the methods. On the other hand, if one method has the best performance for all the test problems, and the performance is satisfactory, then that method is selected and the method selection process is finished on the third step. The selection process is ended also if some methods have acceptable overall performance without any obvious shortcomings. Then one of these methods may be selected. However, it is sometimes recommendable to take a fourth step in the method selection process. On the fourth step, if one method performs well for one type of problems and another method for another type of problems, then one may form a toolbox and use the best method for each type of problem. One may also combine different methods to form hybrids. In [E], the main emphasis was on the test problems and test settings in general. The method selection process is illustrated in Figure 10.

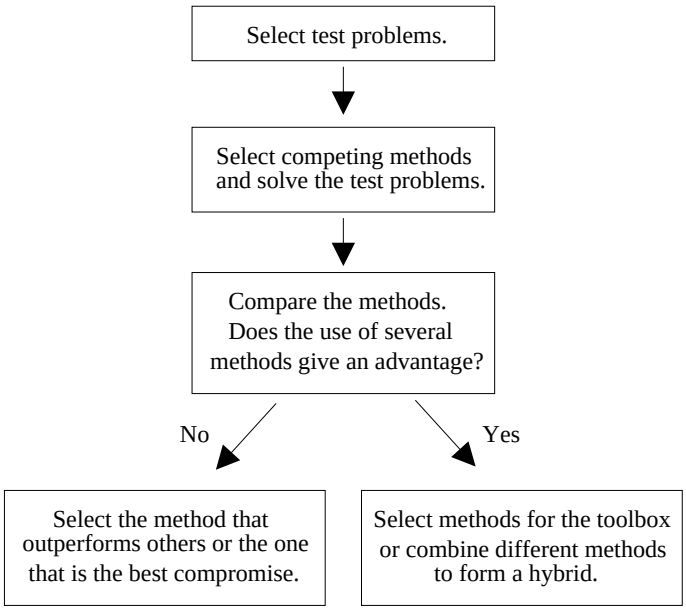

FIGURE 10: Method selection process.

By using numerical examples we pointed out four pitfalls, which all may lead to the selection of a wrong method, that is, a method whose performance is inferior to some other methods in the comparison. The pitfalls were examined in a practical study involving the real-coded genetic algorithm described in [76], multistart method applying Nelder-Mead simplex method and two hybrid methods. We called the first hybrid *genetic algorithm with a local search* GALS. In that hybrid, genetic algorithm was the main optimizer and it was combined with the Nelder-Mead simplex method. Every time the genetic algorithm found a solution, whose objective function value was the best so far, the Nelder-Mead simplex method was started on that solution. In the second hybrid, which we called QSGA, a type of multistart method was the main optimizer, and it provided locally optimal solutions as initial populations for micro genetic algorithm. By micro genetic algorithm we mean a genetic algorithm that has a small population size.

We used these four methods and the results obtained with three state-of-the-art metaheuristics in the literature to illustrate pitfalls in method selection. The first pitfall occurs, if the test problems do not accurately enough correspond to the problems that one will be solving with the algorithm. This was illustrated by solving two test suites using a multistart method applying an SQP solver as a local minimizer. For the first test suite the multistart method was very competitive, even when compared to the state-of-the-art metaheuristics, but for the second test suite its performance collapsed. The main difference between the test suites was the number of local minima.

The second pitfall occurs if not all the performance criteria are considered in the comparison. This was illustrated by first evaluating the efficiency of the methods based on CPU-time alone. When the number of function evaluations was also taken into account, the conclusions changed significantly, even though all the computations were made on the same computer.

The third pitfall occurs, if inefficient stopping criteria are used. This was illustrated by evaluating the performance of the methods after 20,000 and after 100,000 objective function evaluations. Although the genetic algorithm was the best method at both check points with respect to successful runs, some other methods switched places. The best way, of course, would be to use performance profiles [29] or modified performance profiles [3], which record the performance throughout execution of an algorithm.

The fourth pitfall concerns method development. It is important that the performance criteria (and the test problems) are carefully selected already when developing the methods. If incomplete performance criteria are used, they may guide the development off the track. This was illustrated, by developing the hybrid method QSGA using only CPU-time as the efficiency measure and ignoring the number of objective function evaluations. However, many times it is not possible to know all the performance criteria in advance. If the performance criteria are changed, then additional analysis and tests may be required and the necessary adjustments need to be made to the conclusions.

Our study of method comparison was not exhaustive, but we presented a fraction of possible pitfalls. We concluded that method comparison is extremely

difficult. The change of the test suite and the changes in other test settings have strong effects on the conclusion of the comparison. We also pointed out that well-defined and widely used standardized test suites would help to compare different methods. Another important thing would be to report also the results for the problems for which the proposed method did not work so well. For practitioners that piece of information is often just as valuable as knowing when the method performs well.

1.8 Conclusions

The work was devoted to two subareas of global continuous optimization. The first focus area was the hybridization of metaheuristics and the second was the use of structured point sets as initial populations of genetic algorithms. We also included a note on optimization method selection.

The first focus area, hybridization, is widely recognized as one of the most attractive new trends in global optimization in general. In hybridization, elements of existing methods are combined in a way that they best complement each other resulting in a new hybrid method which desirably has the advantages, but not the disadvantages, of the original elements. We gave some general outlines how hybrid methods are formed and presented several new hybrid methods involving simulated annealing and genetic algorithms.

In the hybrids based on simulated annealing, we combined simulated annealing with the proximal bundle method that is a local search method that approximates a decent direction using subgradients. The proximal bundle method can solve even nondifferentiable problems and its efficiency is comparable to gradient-based methods. We evaluated the performance of the methods using 38 test problems from the literature, and the results showed that the hybridization improved both the efficiency and the reliability of original simulated annealing.

We also hybridized a genetic algorithm and the Nelder-Mead simplex method and, again, were able to increase both the efficiency and the reliability of the original methods operating alone. We used the hybrid for training an MLP neural network and evaluated the method by solving a regression problem.

As a general note to hybrids combining different optimization methods, we would like to point out that our experiments in [A], [B] and [E] indicate that, for the test problems used, simple hybridizations worked better than more complex ones. It also became evident that not all combinations of a global and a local search provide any advantage over the original methods.

The second focus area of this work was the use of structured point sets for an initial population of a genetic algorithm. The structured points sets include quasi random sequences, which are used, for example, in numerical integration and simulation, and various spatial processes used, for example, in statistics. If hybridization is defined in a very general way, then the use of structured point sets within an optimization method is a particular type of a hybridization, where all the combined elements are not from optimization methods, but may be any elements

that are available and yet not included in the original method. These methods were here called interdisciplinary hybrids.

Quasi random sequences were the first structured point sets that were applied in global continuous optimization a few decades ago. Since then, the optimization methods have developed considerably and the so-called metaheuristics have become predominant methods for solving complex global optimization problems. However, the use of quasi random sequences in that context has been rather rare. We applied quasi random sequences into initial populations of a genetic algorithm. We also applied two other structured point sets, simple sequential inhibition process and nonaligned systematic sampling process. In addition, we discussed various features of structured point sets.

We expected to see difference in the magnitudes of the variances of objective function values when different initial populations were applied to the genetic algorithm, but the results were quite similar with all the initial populations. We arrived at the conclusion that based on our tests the genetic algorithm used is very robust with respect to initial populations. This is a positive result, but we would like to note that the results presented here are the first results on initial populations of genetic algorithm for continuous optimization and they cannot be directly generalized to other problem sets or other algorithms than the one used in the comparison. In the future, it might be interesting to test the different initial populations with other population-based heuristics that do not rely so much on genetic diversity and recombination.

Finally, we considered method selection problem for global continuous optimization. When selecting an optimization method there are many important issues which are often forgotten in the literature, but which greatly affect the outcome of the comparison. We pointed out four pitfalls that may lead to selecting a method whose performance is inferior to some other methods in the comparison. The pitfalls considered were: carelessly chosen test problems, incomplete performance criteria while developing the method and while testing the method and inefficient stopping criteria during the testing. The pitfalls were illustrated by numerical results of three state-of-the-art metaheuristics in the literature, a genetic algorithm, two multistart methods using the Nelder-Mead simplex method and SQP-solver, respectively, and two hybrids combining genetic algorithms and the Nelder-Mead simplex method. We concluded that it is extremely difficult to compare inherently different methods and that fixing the test settings is crucial.

We would also like to point out some future challenges in global continuous optimization. It seems that different phases in optimization alternate. These phases could be described in familiar optimization terminology as “global” and “local” phases. In the last few decades, the “global phase” has produced concepts such as metaheuristics and hybridization, and local phase has developed these ideas into a variety of different implementations. More recently launched concepts of “global phase” are, for example, taxonomies of hybrid methods, unifying views to metaheuristics [109], interdisciplinary hybrids and hyperheuristics [19]. These emerging areas of “global phase” help to organize and exploit the information and identify similarities both in structure and in ideology of the methods. This way,

the global phase generates new ideas by continually guiding the research into new unexplored areas of optimization.

1.9 Author's contribution

The papers [B]-[E] are based mainly on the author's own ideas. The author has also written them and made the implementations except for the MLP-neural network, the Fortran code for the genetic algorithms and the pseudo and quasi random generators. Professor Antti Penttinen was consulted about the spatial point processes for the paper [D] and he proposed some ways to generate initial populations for genetic algorithms. The methods for paper [A] were implemented by the author's advisers, Professor Kaisa Miettinen and Docent Marko M. Mäkelä, and the paper was written in close collaborations with them. The whole thesis has been closely supervised by the advisers. The supervision has included exchange and refinement of ideas and proof readings. The advisers have also freely shared their professional expertise, knowledge and experiences.

YHTEENVETO (FINNISH SUMMARY)

Tämä väitöskirja käsittelee sitä, kuinka globaalin optimoinnin menetelmiä jatkuvien muuttujien tehtäville voidaan parantaa hybridisointia ja strukturaalisia pistejoukkoja käyttämällä. Optimointi on keskeinen osa ongelmanratkaisua, sillä yleisesti reaalielämän ongelmanratkaisuun kuuluu matemaattinen ja numeerinen mallintaminen sekä mallin simulointi ja optimointi. Ongelmanratkaisun tuloksia voidaan hyödyntää esimerkiksi tuotteiden kehityksessä ja valmistuksessa. Useat reaalimaailman ongelmat voidaan mallintaa jatkuvien muuttujien globaalin optimoinnin tehtäviksi. Numeerisille malleille on usein hyvin vaikeita löytää optimaalista ratkaisua ja siksi tehokkaiden optimointimenetelmien kehittäminen on tarpeen.

Hybridisointia apuna käyttäen, eli menetelmiä yhdistelemällä, on mahdollista kehittää uusia entistä tehokkaampia optimointimenetelmiä. Tässä väitöskirjassa muodostetaan erilaisia hybridimenetelmiä metaheuristiikkojen pohjalta. Tulokset osoittavat, että on mahdollista muodostaa hybridejä, joilla on samat hyvät ominaisuudet kuin yhdistetyissä menetelmissä, mutta joilla ei ole alkuperäisten menetelmien heikkouksia. Työssä yhdistettiin simuloitu jäähdytys ja kimppumenetelmä sekä geneettinen algoritmi ja Nelderin ja Meadin polytooppihaku. Hybridisoinnin tuloksena sekä menetelmien tehokkuus että luotettavuus paranivat. Lisäksi muodostettiin poikkitieteellisiä hybridejä käyttämällä esimerkiksi tilastotieteessä ja numeerisessä integroinnissa käytettyjä pistegeneraattoreita geneettisen algoritmin alkupopulaatioiden muodostamiseen. Pistegeneraattoreiden eri ominaisuuksia tarkasteltiin ja ominaisuuksien vaikutuksia optimointimenetelmän tuloksiin tutkittiin numeerisilla testeillä. Tutkimuksessa kiinnitettiin huomiota myös menetelmien vertailuun. Osoittautui, että optimoinnin testitehtävien valinta ja testien alkuasetelmat pitkälti määrittävät vertailun lopputuloksen.

Kaikkien kehitettyjen menetelmien toimivuus on testattu numeerisilla esimerkeillä. Yleisesti voidaan sanoa, että hybridisointi, eri muodoissaan, voi olla hyvin hyödyllinen työkalu menetelmäkehityksessä.

REFERENCES

[1] Acworth, P., Broadie, M., and Glasserman, P. A comparison of some Monte Carlo and quasi Monte Carlo techniques for option pricing. In *Monte Carlo and Quasi-Monte Carlo Methods 1996*, H. Niederreiter, P. Hellekalek, G. Larcher, and P. Zinterhof, Eds., no. 127 in Lecture notes in statistics. Springer-Verlag, New York, 1998, pp. 1–18.

[2] Al-Sultan, K. S., and Al-Fawzan, M. A. A tabu search Hooke and Jeeves algorithm for unconstrained optimization. *European Journal of Operational Research 103*, 1 (1997), 198–208.

[3] Ali, M. M., Khompatraporn, C., and Zabinsky, Z. B. A numerical evaluation of several stochastic algorithms on selected continuous optimization test problems. *Journal of Global Optimization* (to appear).

[4] Ali, M. M., and Storey, C. Modified controlled random search algorithms. *International Journal of Computer Mathematics 53* (1994), 229–235.

[5] Ali, M. M., and Storey, C. Topographical multilevel single linkage. *Journal of Global Optimization 5*, 4 (1994), 349–358.

[6] Ali, M. M., and Storey, C. Aspiration based simulated annealing algorithm. *Journal of Global Optimization 11*, 2 (1997), 181–191.

[7] Ali, M. M., Törn, A., and Viitanen, S. A numerical comarison of some modified controlled random search algorithms. *Journal of Global Optimization 11*, 4 (1997), 377–385.

[8] Ali, M. M., and Törn, A. Population set-based global optimization algorithms: some modifications and numerical studies. *Computers & Operations Research 31*, 10 (2004), 1703–1725.

[9] Ali, M. M., Törn, A., and Viitanen, S. A direct search variant of the simulated annealing algorithm for optimization involving continuous variables. *Computers & Operations Research 29*, 1 (2002), 87–102.

[10] Androulakis, I. P., Maranas, C. D., and Floudas, C. A. αBB : A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization 7* (1995), 335–363.

[11] Babu, B. V., and Chaurasia, A. S. Pyrolysis of biomass: Improved models for simultaneous kinetics and transport of heat, mass and momentum. *Energy Conversion and Management 45*, 9–10 (2004), 1297–1327.

[12] Banks, J., Carson, J. S., and Nelson, B. L. *Discrete-Event System Simulation*, 2 ed. Prentice Hall, 1996.

[13] Battiti, R., and Tecchiolli, G. The continuous reactive tabu search: Blending combinatorial optimization and stochastic search for global optimization. *Annals of Operations Research 63* (1996), 153–188.

[14] Bazaraa, M. S., and Shetty, C. M. *Nonlinear programming: Theory and algorithms*, 1st ed. John Wiley & Sons, New York, 1979.

[15] Bilbro, G. L., and Snyder, W. E. Optimization of functions with many minima. *IEEE Transactions on Systems, Man, and Cybernetics 21*, 4 (1991), 840–849.

[16] Bonabeau, E., Dorigo, M., and Theraulaz, G. *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, New York, 1999.

[17] Bratley, P., and Fox, B. L. Algorithm 659: Implementing Sobol's quasi-random sequence generator. *ACM Transactions on Mathematical Software 14*, 1 (1988), 88–100.

[18] Bratley, P., Fox, B. L., and Niederreiter, H. Implementation and tests of low-discrepancy sequences. *ACM Transactions on Modeling and Computer Simulation 2*, 3 (1992), 195–213.

[19] Burke, E. K., and Petrovic, S. Recent research directions in automated timetabling. *European Journal of Operational Research 140*, 2 (2002), 266–280.

[20] Chakraborti, N., Kumar, R., and Jain, D. A study of the continuous casting mold using a Pareto-converging genetic algorithm. *Applied Mathematical Modelling 25*, 4 (2001), 287–297.

[21] Chelouah, R., and Siarry, P. Tabu search applied to global optimization. *European Journal of Operational Research 123*, 2 (2000), 256–270.

[22] Chelouah, R., and Siarry, P. Genetic and Nelder-Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research 16*, 2 (2003), 335–348.

[23] Chelouah, R., and Siarry, P. A hybrid method combining continuous tabu search and Nelder-Mead simplex algorithms for the global optimization of multiminima functions. *European Journal of Operational Research* (to appear).

[24] Corne, D., Dorigo, M., and Glover, F., Eds. *New Ideas in Optimization.* McGraw-Hill, 1999.

[25] Cvijović, D., and Klinowski, J. Taboo search: An approach to the multiple minima problem. *Science 267*, 3 (1995), 664–666.

[26] Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons, New York, 2001.

[27] Digalakis, J., and Margaritis, K. Performance comparison of memetic algorithms. *Applied Mathematics and Computation* (to appear).

[28] Diggle, P. J. *Statistical Analysis of Spatial Point Patterns.* Academic Press, London, 1983.

[29] Dolan, E. D., and Moré, J. J. Benchmarking optimization software with performance profiles. *Mathematical Programming 91*, 2 (2002), 201–213.

[30] Dorigo, M., Birattari, M., Blum, C., Gambardella, L. M., Mondada, F., and Stützle, T., Eds. *Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS 2004* (2004), Springer Verlag.

[31] Dorigo, M., and Di Caro, G. The ant colony optimization meta-heuristic. In *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Eds. McGraw-Hill, 1999, pp. 11–32.

[32] Dorigo, M., Di Caro, G., and Sampels, M., Eds. *Ant Algorithms, Third International Workshop, ANTS 2002* (2002), Springer Verlag, Berlin.

[33] Dorigo, M., Maniezzo, V., and Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B* (1996).

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

[35] Dykes, S., and Rosen, B. Parallel very fast simulated reannealing by temperature block partitioning. In *Proceedings of the 1994 IEEE International Conference on Systems, Man, and Cybernetics* (1994), vol. 2, pp. 1914–1919.

[36] Floudas, C. A., and Pardalos, P. M., Eds. *Recent Advances in Global Optimization.* Princeton University Press, Princeton, NJ, 1992.

[37] Fowler, A. Systems modelling, simulation, and the dynamics of strategy. *Journal of Business Research* (2003).

[38] Genetic and evolutionary algorithm toolbox for use with Matlab. http://www.geatbx.com/, June 2004.

[39] Gentle, J. E. *Random Number Generation and Monte Carlo Methods.* Springer-Verlag, New York, 1998.

[40] Glover, F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research 13* (1986), 533–549.

[41] Glover, F. A template for scatter search and path relinking. In *Artificial Evolution*, J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, Eds. Springer-Verlag, New York, 1998, pp. 13–54.

[42] Glover, F., and Kochenberger, G. A., Eds. *Handbook of Metaheuristics.* Kluwer Academic Publishers, Boston, 2003.

[43] Glover, F., and Laguna, M. *Tabu Search.* Kluwer Academic Publishers, Dordrecht, 1997.

[44] Glover, F., Laguna, M., and Martí, R. Fundamentals of scatter search and path relinking. *Control and Cybernetics 39*, 3 (2000), 653–684.

[45] Goffe, W. L., Ferrier, G. D., and Rogers, J. Global optimization and statistical functions with simulated annealing. *Journal of Econometrics 60*, 1–2 (1994), 65–99.

[46] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, New York, 1989.

[47] Hanagandi, V., and Nikolaou, M. A hybrid approach to global optimization using a clustering algorithm in a genetic search framework. *Computers & Chemical Engineering 22*, 12 (1998), 1913–1925.

[48] Hedar, A.-L., and Fukushima, M. Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization. *Optimization Methods and Software 17*, 5 (2002), 891–912.

[49] Hellekalek, P., and Larcher, G., Eds. *Random and Quasi-Random Point Sets.* Springer-Verlag, New York, 1998.

[50] Holland, J. H. *Adaption in Natural and Artificial Systems.* MIT Press, Cambridge, 1975.

[51] Hooke, R., and Jeeves, T. A. "Direct search" solution of numerical and statistical problems. *Journal of the ACM 8*, 2 (1961), 212–229.

[52] Horst, R., and Pardalos, P. M., Eds. *Handbook of Global Optimization.* Kluwer Academic Publishers, Dordrecht, 1995.

[53] Ingber, L. Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling 18*, 11 (1993), 29–57.

[54] Jain, A. S., and Meeran, S. Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research 113*, 2 (1999), 390–434.

[55] Kelley, C. T. *Iterative Methods for Optimization.* SIAM, Philadelphia, 1999.

[56] Kingsley, M. C. S., Kanneworff, P., and Carlsson, D. M. Buffered random sampling: a sequential inhibited spatial point process applied to sampling in a trawl survey for northern shrimp pandalus borealis in west Greenland waters. *ICES Journal of Marine Science 61*, 1 (2004), 12–24.

[57] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science 220*, 4598 (1983), 671–680.

[58] Kiwiel, K. C. Proximity control in bundle methods for convex nondifferentiable optimization. *Mathematical Programming 46*, 1 (1990), 105–122.

[59] Laguna, M. Scatter search. In *Handbook of Applied Optimization*, P. M. Pardalos and M. G. C. Resende, Eds. Oxford University Press, New York, 2002, pp. 183–193.

[60] Laguna, M., and Martí, R. Neural network prediction in a system for optimizing simulations. *IIE Transactions 34*, 3 (2002), 273–282.

[61] Laguna, M., and Martí, R. *Scatter Search: Methodology and Implementations in C*. Kluwer Academic Publishers, Boston, 2003.

[62] Lécot, C. Quasi-randomized numerical methods for systems with coefficients of bounded variation. *Mathematics and Computers in Simulation 55* (2001), 113–121.

[63] Lei, G. Adaptive random search in quasi-Monte Carlo methods for global optimization. *Computers & Mathematics with Applications 43*, 6–7 (2002), 747–754.

[64] Lemaréchal, C. Nondifferentiable optimization. In *Optimization*, G. L. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, Eds. North-Holland, Amsterdam, 1989, pp. 529–572.

[65] Lemieux, C., and L'Ecuyer, P. On selection criteria for lattice rules and other quasi-Monte Carlo point sets. *Mathematics and Computers in Simulation 55*, 1–3 (2001), 139–148.

[66] Liu, H.-L. A hybrid AI optimization method applied to industrial processes. *Chemometrics and Intelligent Laboratory Systems 45*, 1–2 (1999), 101–104.

[67] Maaranen, H. On Global Optimization with Aspects to Method Comparison and Hybridization. Licentiate thesis, Department of Mathematical Information Technology, University of Jyväskylä, 2002.

[68] Maaranen, H., Miettinen, K., and Mäkelä, M. M. Using quasi random sequences in genetic algorithms. In *Optimization and Inverse Problems in Electromagnetism*, M. Rudnicki and S. Wiak, Eds. Kluwer Academic Publishers, Dordrecht, 2003, pp. 33–44.

[69] Mäkelä, M. M., and Neittaanmäki, P. *Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control*. World Scientific Publishing Co., Singapore, 1992.

[70] Martí, R., and El-Fallahi, A. Multilayer neural networks: an experimental evaluation of on-line training methods. *Computers & Operations Research 31*, 9 (2004), 1491–1513.

[71] Mendoza, N. E., Chen, Y.-W., Nakao, Z., Adachi, T., and Masuda, Y. A real multi-parent tri-hybrid evolutionary optimization method and its application in wind velocity estimation from wind profiler data. *Applied Soft Computing 1*, 3 (2001), 225–235.

[72] Merz, P., and Freisleben, B. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In *Congress of Evolutionary Computation (CEC'99)* (1999), IEEE Press, Piscataway, NJ, pp. 2063–2070.

[73] Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin, 1994.

[74] Michalewicz, Z. Evolutionary computations and heuristics. In *Meta-Heuristics: Theory and Applications*, I. H. Osman and J. P. Kelly, Eds. Kluwer Academic Publishers, Boston, 1996, pp. 37–52.

[75] Michalewicz, Z., and Schoenauer, M. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation 4*, 1 (1996), 1–32.

[76] Miettinen, K., Mäkelä, M. M., and Toivanen, J. Numerical comparison of some penalty-based constraint handling techniques in genetic algorithms. *Journal of Global Optimization 27*, 4 (2003), 427–446.

[77] Morokoff, W. J., and Caflisch, R. E. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing 15*, 6 (1994), 1251–1279.

[78] Moscato, P. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Tech. Rep. Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA, 1989.

[79] Nelder, J. A., and Mead, R. A simplex method for function minimization. *The Computer Journal 7* (1965).

[80] Niederreiter, H. Quasi-Monte Carlo methods for global optimization. In *Proceedings of the 4th Pannonian Symposium on Mathematical Statistics* (1983), W. Grossmann, G. Pflug, I. Vincze, and W. Wertz, Eds., pp. 251–267.

[81] Niederreiter, H. *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM, Philadelphia, 1992.

[82] Nix, A. E., and Vose, M. D. Modeling genetic algorithms with Markov chains. *Annals of Mathematics and Artificial Intelligence 5*, 1 (1992), 79–88.

[83] Onwubolu, G. D., and Babu, B. V. *New Optimization Techniques in Engineering*, vol. 141 of *Studies in Fuzziness and Soft Computing*. Springer Verlag, New York, 2004.

[84] Özdamar, L., and Demirhan, M. Experiments with new stochastic global optimization search techniques. *Computers & Operations Research 27*, 9 (2000), 841–865.

[85] Pardalos, P. M., and Romeijn, H. E., Eds. *Handbook of Global Optimization*, vol. 2. Kluwer Academic Publishers, Boston, 2002.

[86] Pardalos, P. M., Romeijn, H. E., and Tuy, H. Recent developments and trends in global optimization. *Journal of Computational and Applied Mathematics 124*, 1–2 (2000), 209–228.

[87] Pedgen, C. D., Shannon, R. E., and Sadowski, R. P. *Introduction to Simulation using Siman*. McGraw-Hill, New York, 1995.

[88] Pirlot, M. General local search methods. *European Journal of Operational Research 92*, 3 (1996), 493–511.

[89] Press, W. H., and Teukolsky, S. A. Quasi- (that is, sub-) random numbers. *Computers in Physics 3*, 6 (1989), 76–79.

[90] Preux, P., and Talbi, E.-G. Towards hybrid evolutionary algorithms. *International Transactions in Operational Research 6*, 6 (1999), 557–570.

[91] Price, K. V., Storn, R., and Lampinen, J. *Differential Evolution: A Practical Approach to Global Optimization*. Natural Computing Series. Springer Verlag, to appear.

[92] Price, W. L. Global optimization by controlled random search. *Journal of Optimization Theory and Applications 40*, 3 (1983), 333–348.

[93] Reeves, C. Genetic algorithms. In *Handbook of Metaheuristics*, F. Glover and G. A. Kochenberger, Eds. Kluwer Academic Publishers, Boston, 2003, ch. 3, pp. 55–82.

[94] Resende, M. G. C., and de Sousa, J. P., Eds. *Metaheuristics: Computer Decision-Making*. Kluwer Academic Publishers, Dordrecht, 2003.

[95] Salami, M., and Hendtlass, T. A fast evaluation strategy for evolutionary algorithms. *Applied Soft Computing 2*, 3 (2003), 156–173.

[96] Salhi, S., and Queen, N. M. A hybrid algorithm for identifying global and local minima when optimizing functions with many minima. *European Journal of Operational Research 155*, 1 (2004), 51–67.

[97] Samanta, B. Artificial neural networks and genetic algorithms for gear fault detection. *Mechanical Systems and Signal Processing 18*, 5 (2004), 1273–1282.

[98] Schoen, F. Random and quasi-random linkage methods in global optimization. *Journal of Global Optimization 13*, 4 (1998), 445–454.

[99] Siarry, P., and Berthiau, G. Fitting of tabu search to optimize functions of continuous variables. *International Journal for Numerical Methods in Engineering 40*, 13 (1997), 2449–2457.

[100] Siarry, P., Berthiau, G., Durdin, F., and Haussy, J. Enhanced simulated annealing for globally minimizing functions of many-continuous variables. *ACM Transactions on Mathematical Software 23*, 2 (1997), 209–228.

[101] SIMANN – Fortran simulated annealing code used in [45]. Available at http://wueconb.wustl.edu/~goffe/, June 2004.

[102] Snyder, W. C. Accuracy estimation for quasi-Monte Carlo simulations. *Mathematics and Computers in Simulation 54*, 1–3 (2000), 131–143.

[103] Sobol', I. M. On quasi-Monte Carlo integrations. *Mathematics and Computers in Simulation 47*, 2–5 (1998), 103–112.

[104] Sobol', I. M., and Bakin, S. G. On the crude multidimensional search. *Journal of Computational and Applied Mathematics 56*, 3 (1994), 283–293.

[105] Sobol, I. M., and Tutunnikov, A. V. A varience reducing multiplier for Monte Carlo integrations. *Mathematical and Computer Modelling 23*, 8–9 (1996), 87–96.

[106] Sotiropoulos, D. G., Stavroppulos, E. C., and Vrahatis, M. N. A new hybrid genetic algorithm for global optimization. *Nonlinear Analysis, Theory, Methods & Applications 30*, 7 (1997), 4529–4538.

[107] Storn, R., and Price, K. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Tech. rep., International Computer Science Insitute (ICSI), 1995.

[108] Storn, R., and Price, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization 11*, 4 (1997), 341–359.

[109] Taillard, . D., Gambardella, L. M., Gendreau, M., and Potvin, J.-Y. Adaptive memory programming: A unified view of metaheuristics. *European Journal of Operational Research 135*, 1 (2001), 1–16.

[110] Talbi, E.-G. Taxonomy of hybrid metaheuristics. *Journal of Heuristics 8*, 5 (2002), 541–564.

[111] Törn, A., and Viitanen, S. Iterative topographical global optimization. In *State of the art in global optimization*, C. A. Floudas and P. M. Pardalos, Eds. Kluwer Academic Publishers, 1996, pp. 353–363.

[112] Törn, A., and Žilinskas, A. *Global Optimization*. Springer-Verlag, Berlin, 1989.

[113] Trafalis, B., and Kasap, S. A novel metaheuristics approach for continuous global optimization. *Journal of Global Optimization 23*, 2 (2002), 171–190.

[114] Trelea, I. C. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters 85*, 6 (2003), 317–325.

[115] Tuffin, B. On the use of low discrepancy sequences in Monte Carlo methods. *Monte Carlo Methods and Applications 2*, 4 (1996), 295–320.

[116] Vanderbilt, D., and Louie, S. G. A Monte Carlo simulated annealing approach to optimization over continuous variables. *Journal of Computational Physics 56* (1984), 259–271.

[117] Victoire, T. A. A., and Jeyakumar, A. E. Hybrid PSO-SQP for economic dispatch with valve-point effect. *Electric Power Systems Research 71*, 1 (2004), 51–59.

[118] Voss, S., Martello, S., Osman, I. H., and Roucairol, C., Eds. *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Kluwer Academic Publishers, Boston, 1999.

[119] Wang, P. P., and Chen, D.-S. Continuous optimization by a variant of simulated annealing. *Computational Optimization and Applications 6*, 1 (1996), 59–71.

[120] Weisstein, E. W. "circle packing". From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/CirclePacking.html, September 2004.

[121] Wetter, M., and Wright, J. A comparison of deterministic and probabilistic optimization algorithms for nonsmooth simulation-based optimization. *Building and Environment 39*, 8 (2004), 989–999.

[122] Wright, A. H., and Zhao, Y. Markov chain models of genetic algorithms. In *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference* (1999), W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, Eds., Morgan Kauffman, SanMateo, California, pp. 734–741.

[123] Yen, J., Liao, J. C., Lee, B., and Randolph, D. A hybrid approach to modeling metabolic systems using a genetic algorithm and simplex method. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 28*, 2 (1998), 173–191.

[124] Yiu, K. F. C., Liu, Y., and Teo, K. L. A hybrid descent method for global optimization. *Journal of Global Optimization 28*, 2 (2004), 229–238.

[125] Yong, L., Lishan, K., and Evans, D. J. The annealing evolution algorithm as function optimizer. *Parallel Computing 21*, 3 (1995), 389–400.

[126] Yu, H., Fang, H., Yao, P., and Yuan, Y. A combined genetic algorithm/simulated annealing algorithm for large scale system energy integration. *Computers & Chemical Engineering 24*, 8 (2000), 2023–2035.