Performance index and meta-optimization of a direct search optimization method

Petter Krus and Johan Ölvander

Submitted to Journal of Engineering Optimization

Performance Index and Meta Optimization of a Direct Search Optimization Method

P. KRUS* and J. ÖLVANDER

Department of Management and Engineering
Linköping University
SE-59049 Linköping, Sweden

*Corresponding author. Email: Petter.Krus@liu.se

Abstract

Design optimization is becoming an increasingly important tool for design, often using simulation as part of the evaluation of the objective function. A measure of the efficiency of an optimization algorithm is of great importance when comparing methods. The main contribution of this paper is the introduction of a singular performance criteria, the *Entropy Rate Index*, based on Shannon's information theory, taking both reliability and rate of convergence into account. It can also be used to characterize the difficulty of different optimization problems. Such a performance criterion can also be used for optimization of the optimization algorithms itself.

In this paper the Complex-RF optimization method is described and its performance evaluated and optimized using the established performance criterion.

Finally, in order to be able to predict the resources needed for optimization an *objective function temperament factor*, is defined that indicates the degree of difficulty of the objective function.

Keywords: Design optimization, performance index, information entropy, direct search optimization, Complex method

1. Introduction

The rapid development in simulation methods and the general increase in hardware performance imply that design methods based on different kinds of numerical optimization for system design are becoming increasingly important. Numerical optimization methods require that the objective function to be evaluated (using simulation) a great many times, but they are very attractive since they can be used for optimization based on simulation of complete, non-linear systems, and do not rely on grossly simplified models, where gradients are available, as more analytical methods do. Work in this area has shown that simulation based optimization can be used both for parameter optimization and for component sizing; see for example Krus et al. (1993).

There are many situations where very little information is available regarding the nature of the objective function, e.g. gradients cannot be obtained explicitly and constraints are implicit. This is particularly true when evaluation of the objective function relies on simulation of dynamic systems. In these situations direct search methods are very attractive, and in this paper the focus is on a specific direct search method, the Complex method, see Box (1965). This method, and some modified versions of it, are promising candidates for direct search optimization, and are analyzed in more detail in this paper.

One issue when developing optimization methods is how to evaluate their performance. Although it may sound deceptively simple, it is a very multifaceted problem. Perhaps the most obvious difficulty is that the performance of a particular optimization method is highly problem-dependent. This means that for every class of objective functions it is relevant to test a range of methods to find the most suitable one. Furthermore, even with a given problem, there are several ways to measure performance, the most important being the number of evaluations needed to converge and the probability of reaching the true optima. The need for efficient direct search methods constitutes the motivation for this research. In order to increase the efficiency of any optimization method, the algorithm itself should be optimized, and hence there is a need to establish a performance index to measure the performance of the optimization algorithm itself.

1.1 Performance measures

The majority of investigations where performance of methods is presented and compared, such as for example Afsar (2008), Liu (2009), Schutte (2005), Wang (2007), Younis and Dong (2010), and Zahara (2008), use the convergence plot of the objective functions (and sometimes parameters). In addition, the number of objective function evaluations is displayed, as well as how close the objective function is to the true optimum. Reklaitis et al. (1983) list a number of criteria such as CPU time, accuracy in the found solution, number of iterations, reliability, number of function evaluations, user friendliness, and storage requirements. Of these the relevant are really CPU time, accuracy, robustness and user friendliness, since number of iterations and function evaluations are merely other measures of CPU time. However, if the measure should be machine independent, it is really the number of function evaluations that is important, since most of the CPU time is likely to be in the function evaluations, rather than in the calculations in the algorithm itself. This is especially true for simulation based optimization. User-friendliness implies that the method should have acceptable performance for a wide range of optimization problems, with a minimum of tuning of the algorithm. Robustness can be defined as the probability that the true optimum will be found when the algorithm is applied to a problem. Khompatraporn et al (2005) also describe various problems associated with making comparisons, such as test problem selection, and how different stop criteria influence a method's performance. Although performance measures based on the fact that number of function evaluations is of interest are presented, the reliability is not so often presented. In Liao and Luus (2006), however, it is also shown how many times, from a set of trials, the optimization converge on the true solution. This was also the case in Manetsch and Cabrera (1991), In addition they also looked into the effect of using several processors in parallel. Looking at this literature there is not a common established best practice for how to present performance in a way so that it is easily comparable between different papers and authors.

1.2 Meta-optimization

The term meta-optimization could be used to describe the formal optimization of an optimization algorithm (compare with a meta-model being a model of a model). Keane (1995) establish a criterion only related to the objective function value and the number of evaluations, which is subsequently used to tune a Genetic algorithm. In this way there is another layer of optimization where the optimization algorithm itself is optimized, hence a meta-optimization. In a similar manner Pedersen and Chipperfield (2009) used meta-optimization to tune parameters in a particle swarm method. Smit and Eiben (2009) use different methods to tune parameters in an evolutionary algorithm, including evolutionary algorithm itself. In Krus and Andersson (2003) used meta-optimization of a modified version of the Complex method using a criterion combining both number of objective function evaluations and the probability of finding the correct optimum

1.3 Objective function characterization

Finally, in order to be able to predict the resources needed for optimization, it would be a great advantage if the difficulty of an optimization problem could be characterized. In the present situation this is expressed only in general terms, such that a problem is convex, multi modal or NP-hard.

It can therefore be concluded that there is a lack of an agreed upon formulation that unifies convergence rate and reliability into a measure that can be used for comparison between different studies of optimization methods and problems, and there is also a lack of quantification of the difficulty of different optimization problems.

The remainder of the paper has the following structure. First Shanon's information theory is described in the context of a generic optimization process. Thereafter the Complex method is explained in detailed together with its convergence properties. The proposed performance index is then developed and applied to a set of test functions. Next, meta-optimization of the Complex method is performed using the Complex algorithm itself and the proposed performance index. Finally, a difficulty index for objective functions is presented.

2. Information theory and design optimization

In Pierce (1978) and subsequently in Jaynes (1981) it is described how information theory can be applied to the search (surveillance) problem. It is therefore used to optimize the search strategy. Optimization in itself is a kind of search, and can be regarded as a learning process where information about the parameter values is gained as the optimization progresses. It is therefore natural to investigate the optimization process from an information-theoretical point of view. Information theory of communication was founded by C.E. Shannon with his paper "*A Mathematical Theory of Communication*" (1948). This is basically a probabilistic approach where information theory is related to thermodynamics with the introduction of the entropy as a measure of information.

An interesting aspect is to study the amount of information gained in each evaluation. Shannon defined the differential information entropy for continuous signals as

$$H = -\int_{-\infty}^{\infty} p(x)\log_2\left(p(x)\right)dx \tag{1}$$

This gives one measure of information. Here $p(x)$ is the probability density function. This expression is not dimensionless, unless x is dimensionless, since the probability density function has the unit of the inverse of x. If not, the differential entropy of the probability density function $p(x)$ needs to be related to another distribution $m(x)$. The result is called the Kullback-Leibler divergence (Kullback and Leibler, 1951) from the distribution $m(x)$. This is the relative entropy and it is defined as

$$H_{rel} = \int_{-\infty}^{\infty} p(x)\log_2\left(\frac{p(x)}{m(x)}\right)dx \tag{2}$$

It can also be generalized to any dimensionality.

$$H_{rel} = \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} p(x_1\ldots x_n)\log_2\left(\frac{p(x_1\ldots x_n)}{m(x_1\ldots x_n)}\right)dx_1\cdots dx_n \tag{3}$$

This can be written in a more compact form as

$$H_{rel} = \int_{D_{Inf}} p(\mathbf{x}) \log_2 \left(\frac{p(\mathbf{x})}{m(\mathbf{x})} \right) d\mathbf{x} \tag{4}$$

Here $m(\mathbf{x})$ is the a priori distribution of the design parameters in the unbounded design space D_{Inf}. $p(\mathbf{x})$ is a posterior distribution of design parameters and H_{rel} is the design information entropy needed for the corresponding reduction of uncertainty and has the unit *bits*. For optimization $p(\mathbf{x})$ would represent the uncertainty of the optimization variables after the optimization and $m(x)$ would represent the original design space of the optimization variables, i.e. the variable limits.

A rectangular distribution of $m(x)$ in the bounded interval $x \in [x_{\min}, x_{\max}]$, with $x_R = x_{\max} - x_{\min}$, would mean that the design space $m(x)$ is a space of equal possibilities, where no particular region can be considered more likely than another. Other distributions can also be considered but they can always be mapped on a rectangular distribution by transforming the design space, which can be very useful (this also includes infinite distributions), for design optimization. Equation (2) can then be rewritten as

$$H_x = H_{rel} = \int_{x_{\min}}^{x_{\max}} p(x) \log_2 \left(p(x) x_R \right) dx \tag{5}$$

H_x is here used to indicate relative information entropy related to a rectangular distribution and has the unit *bits*. For the multidimensional case it will be

$$H_x = \int_{x_{1,\min}}^{x_{1,\max}} \cdots \int_{x_{n,\min}}^{x_{n,\max}} p(x_1, \ldots, x_n) \log_2 \left(p(x_1, \ldots, x_n) \prod_{i=1}^{n} x_{Ri} \right) dx_1 \cdots dx_n \tag{6}$$

This can also be written in a more compact form as

$$H_x = \int_D p(\mathbf{x}) \log_2 (p(\mathbf{x}) S) d\mathbf{x} \tag{7}$$

where D is the design space. H_x is defined as the *design information entropy* where design $\mathbf{x}$ has a posteriori distribution $p(\mathbf{x})$ within the design space D. S is the size of the design space and is defined as

$$S = \int_D \mathbf{x} d\mathbf{x} \tag{8}$$

If the range $[x_{\min}, x_{\max}]$ of one variable is divided into equal parts Δx, but where it is known that the value is in a particular region $[x_{0L}, x_{0H}]$, the probability density distribution will be

$$\begin{aligned} p(x) &= \frac{1}{\Delta x} : x \in [x_{0L}, x_{0H}] \\ p(x) &= 0 : x \in [x_{\min}, x_{\max}] \setminus [x_{0L}, x_{0H}] \end{aligned} \tag{9}$$

where

$$\begin{aligned} x_R &= x_{\max} - x_{\min} \\ \Delta x &= x_{0L} - x_{0H} \end{aligned} \tag{10}$$

This yields the information content (in bits) for that variable as:

$$H_x = \int_{x_{0L}}^{x_{0H}} \frac{1}{\Delta x} \log_2 \left(\frac{x_R}{\Delta x} \right) dx = \log_2 \frac{x_R}{\Delta x} = \log_2 \frac{1}{\delta_x} = -\log_2 \delta_x \tag{11}$$

where Δx is the uncertainty of the variable and x_R its design span. δ_x is introduced as the relative uncertainty in parameters. The same expression holds if the probability distribution is normally distributed, in which case

$$\delta_x = \frac{2\sigma_x}{x_R} \tag{12}$$

Here σ_x is the standard deviation in x. In the following it is assumed that the uncertainty can be described by δ_x.

3. The Complex Algorithm

There are basically two families of optimization methods used in engineering. The gradient methods are widely used and are suitable for problems where the gradient of the objective function can be calculated explicitly at each point. This is for example the case in many structure optimization applications. The other group is the non-gradient methods. These methods do not rely explicitly on gradient information in each point. This also includes "gradient" methods where the gradient is calculated through some differentiation scheme. These methods are therefore of more general use, since gradient information is not generally available, especially if parts of the objective function are evaluated using simulation of non-linear systems. A modified version of the original Complex method by Box (1965) has been found to be one of the simplest and most easy to use methods. Among other things it has been used for simulation-based system optimization of hydraulic systems, see for example Krus et al (1993), Petterson et al. (2009) and Gavel et al. (2006). The implementation shown here has also been described in Krus and Gunnarsson (1993), where it was used for parameter optimization of a control system for a hydraulic crane where real experiments were used for function evaluation.

In the Complex method the word *complex* refers to a geometric shape with $m \geq n+1$ points in an n-dimensional space. These m points are known as vertices of the Complex. The initial Complex is generated by sampling random points within the variable limits. If the implicit constraints are not fulfilled, a new point is generated until they are.
The objective function is evaluated at each point. The worst point, i.e. the point with the highest function value for a minimization problem, is replaced by a point reflected in the centroid of the remaining points by a factor α.

$$x_{ij}^{new} = x_{ic} + \alpha\left(x_{ic} - x_{ij}^{worst}\right) \tag{13}$$

The centroid is calculated as

$$x_{ic} = \frac{1}{m-1}\left(\sum_{j=1}^{m} x_{ij} - x_{ij}^{worst}\right) \tag{14}$$

Box recommends $\alpha = 1.3$. If a point repeats as the highest value in consecutive trials, it is moved one half the distance towards the centroid of the remaining points, in this case

$$x_{ij}^{new'} = x_{ic} + \left(x_{ic} - x_{ij}^{new}\right)/2 \tag{15}$$

The Complex-RF optimization method described in the remainder of this paper is a modified version of the Complex method.

3.1 Convergence rate

Assuming that all points in the Complex are located at approximately the same distance from the optimum, the maximum contraction speed of the Complex method in each objective function evaluation can be estimated as (Krus and Gunnarsson, 1993)

$$\frac{\Delta x_{k+1}}{\Delta x_k}=\left(\frac{\alpha}{2}\right)^{\frac{1}{2m}} \tag{16}$$

The number of parameters *m* in the Complex is a function of the number of independent variables $m=\kappa n$ where typically $1.5<\kappa<2$.
Here Δx_k is the spread of the Complex parameter set, at a particular evaluation number *k*. Expressed as a function of the original spread Δx_0, this gives

$$\varepsilon=\frac{\Delta x_{k+1}}{\Delta x_0}=\left(\frac{\alpha}{2}\right)^{\frac{k}{2\kappa n}} \tag{17}$$

This means that the number of calculations needed to reduce the spread of all variables down to at least ε of the original spread can be estimated by

$$k=\frac{2\kappa}{\log\left(\frac{2}{\alpha}\right)}n\log\left(\varepsilon\right) \tag{18}$$

From this simple relationship follows that the number of evaluations needed to reduce a Complex by a certain amount is linearly dependent on the number of points in the Complex. In reality the objective function can be much more complicated than assumed here, but this estimate gives a lower bound to the number of evaluations necessary and a fair description of the behaviour near optimum.

The increase in information entropy for each objective function evaluation is obtained using eq. (11) and eq. (16) and can be written

$$\Delta H_x=-n\log_2\left(\frac{\Delta x_{k+1}}{\Delta x_k}\right)=-n\log_2\left(\frac{\alpha}{2}\right)^{\frac{1}{2\kappa n}} \tag{19}$$

where the multiplication by *n* comes from the fact that all *n* variables gain information. This can be simplified to

$$\Delta H_x=-\log_2\left(\frac{\alpha}{2}\right)^{\frac{1}{2\kappa}} \tag{20}$$

which with α=1.3 and k =2 yields

$$\Delta H_x=-\log_2\left(\frac{1.3}{2}\right)^{\frac{1}{4}}=0.155 \tag{21}$$

This means that the system gains on average 0.155 bits of information for each evaluation. Note that this is independent of the number of optimization variables. However, for more optimization variables, it takes longer to converge to the same degree, since more information is needed. This represents an estimate of the amount of information gained in each function evaluation for a simple optimization problem. For more difficult problems, however, this can be expected to be considerably less.

3.2 The Complex-RF method

In order to enhance the performance of the original Complex method some modifications are useful. One is to introduce more randomization in the search. This avoids premature collapse of the method.

$$x_{ij}^{new} = x_{ic} + \alpha\left(x_{ic} - x_{ij}^{worst}\right) + x_{r,i} \tag{22}$$

where $x_{r,i}$ is a random number that is calculated as

$$x_{r,i} = (r_1 - 0.5)\max\left(\delta_{x,l}\right)(h_i - g_i)\beta \tag{23}$$

where r_1 is a random number in the interval [0,1], and β is a randomization factor, a constant, with a typical value in the interval [0,1.5]. Furthermore, the relative spread in a parameter is

$$\delta_{x,i} = \frac{x_{i,\max} - x_{i,\min}}{x_{\max} - x_{\min}} \tag{24}$$

where $x_{i,max}$ and $x_{i,min}$ are the maximum and minimum values of x_i in the current Complex. In this way a random value is added to each parameter value that is dependent on the maximum spread in any parameter and scaled with the design range of the specific parameter.

Another modification involves what happens if a point repeats as the highest value in consecutive trials. Instead of just moved halfway towards the centroid of the Complex, it is also mirrored in the centroid. This handles implicit constraints or sharp edges in the objective function better, since it avoids a premature collapse at the edge.

$$x_{ij}^{new'} = x_{ic} - \left(x_{ic} - x_{ij}^{new}\right)\big/2 + x_{r,ij} \tag{25}$$

It is also possible to include a forgetting factor, which ensures that the Complex is made up predominantly with recent parameter sets. This is necessary if the objective function varies over time. In this case old objective function values become increasingly unreliable and should be replaced by new ones. This is particularly true if the optimization is to be used to optimize parameters in a real process, in which case there may be drift in the parameters of the physical system.

Furthermore, if the objective function is stationary but noisy, (i.e. there are local variations in the objective function between points close to each other in parameter space) this is indistinguishable from time-dependent noise since the probability of having a parameter set return to exactly the same position (and thus the same objective function value) is very small.

In each time step all stored objective function values in the Complex are subtracted with the value f_{0e}:

$$f_i = f_i^{old} - f_{0e} \tag{26}$$

where f_{0e} is calculated as:

$$f_{0e} = (f_{\min} - f_{\max})\left(1 - \left(\frac{\alpha}{2}\right)^{\frac{-\gamma}{m}}\right) \tag{27}$$

After each object value evaluation, all stored objective function evaluations in the Complex are decreased with this value f_{0e}. Using this approach, an objective function value is gradually reduced until it is replaced. The value of γ can then be chosen to yield the desired properties. The proper selection will be discussed later.

4. Performance index of an optimization algorithm

The objective of any optimization algorithm is to gather information about the optimum solution. This can be expressed as reducing the uncertainty by a certain amount and thus increasing the amount of information as in

(20). The cost is assumed to be simply proportional to the number of evaluations, *N*, needed to converge at the optimum. An alternative would be to use the clock time needed to converge. In this case the mean value, N_m, of a set of different optimization runs will be used. A natural measure of merit when evaluating an optimization algorithm would thus be

$$\phi = \frac{H_x}{N_m} = -\frac{n \log_2 \varepsilon_x}{N_m} \tag{28}$$

Here, the tolerance ε_x is used as a measure of the final degree of contraction. Alternatively the actual values of the variable spread can be used. First, a very simple test function is used, which is a simple unimodal hump, described by (32).

$$f_1(x_1, x_2) = \sin(\pi x_1)\sin(\pi x_2) \tag{29}$$

where

$$x_i \in [0, 0.7], \;\; i = 1, 2$$

Running one hundred optimizations with the Complex method where the stop criterion is set to ε=0.001 and taking the average value of the performance index yields

$$\phi = n\frac{\log_2 \varepsilon}{N_m} = -2\frac{\log_2 0.001}{99} = 0.2 \tag{30}$$

This is a little more than the theoretical estimate of 0.155. Typical convergence behaviour is shown in Figure 1, where the maximum relative spread in *x* in each evaluation step is plotted. Figure 2 shows an estimation of the information expressed as

$$\hat{H}_x = -n\log_2\left(\max\left(\delta_{x,i}\right)\right) \tag{31}$$

This represents an estimate of the information gathered. In the same graph a straight line representing the theoretical convergence in equation (24) is also shown.

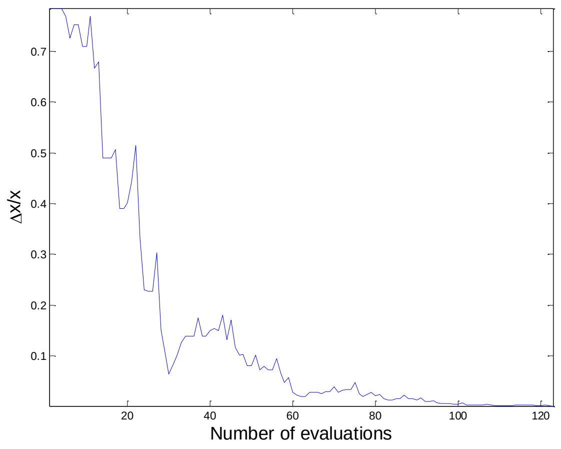

Figure 1. Convergence of optimization variables, i.e. the maximum relative spread δ_x of all (both) variables.

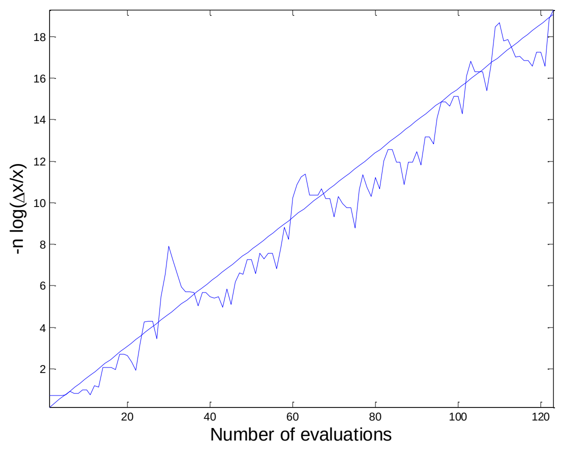

Figure 2. The convergence expressed as –n log2 max(δ_x), and a straight line corresponding to the theoretical convergence rate.

It should be noted that equation (24) is not the only way to define the estimate. Another way would definitely be to add the estimated information entropy of each variable and sum them.

$$\hat{H}_x = -\sum_{i=1}^{n} \log_2 \delta_{x,i} \tag{32}$$

This function will, however, be much noisier since it essentially tracks the variable with the least spread, but the choice is to some extent a matter of taste.

A more challenging test function is the so-called “Rosenbrock’s banana”, Rosenbrock (1960), see equation (26). This is a function that exhibits widely varying gradients in different directions. This means that the search algorithm finds the ridge but fails to detect the gradient along the ridge and consequently to proceed to the optimum.

$$f_2\left(x_1, x_2\right) = 100\left(x_1^2 - x_2\right)^2 + \left(1 - x_1\right)^2 \tag{33}$$

where

$$x_i \in [-2, 2], \quad i = 1, 2$$

This test function is so tricky that convergence at the optimum solution is not guaranteed for a finite stop criterion. This can be seen in Figure 3, where the convergence in parameters for one optimization run is shown. It can be seen that the Complex first converges at around 25 evaluations, when it homes in on a part of the ridge. It then expands again, moving along the ridge and finally converging on the optimum. A similar behaviour can be seen where the estimate of the convergence is plotted along with the theoretical convergence rate. In this case, the convergence is on average much slower, although it may have local bursts of quicker convergence, approaching the theoretical speed. It must be stressed that there is a difference between estimated information and the actual information. Even though the estimated information can be reduced locally during optimization this only reflects the fact that the uncertainty was underestimated: the optimizer was converging towards a false optimum. This means that the estimated information entropy is always higher than the actual information entropy.

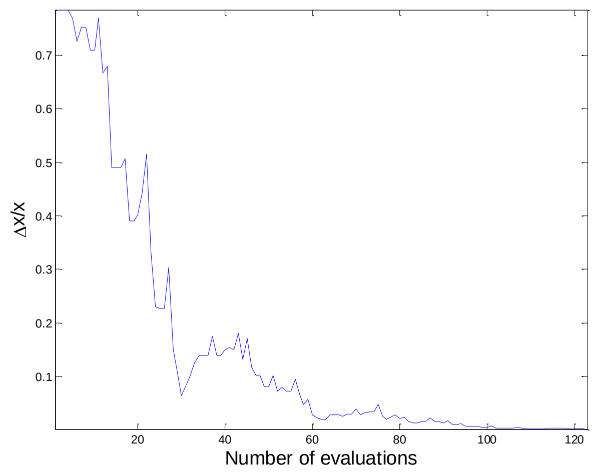

Figure 3. The convergence rate in parameters for the banana function.

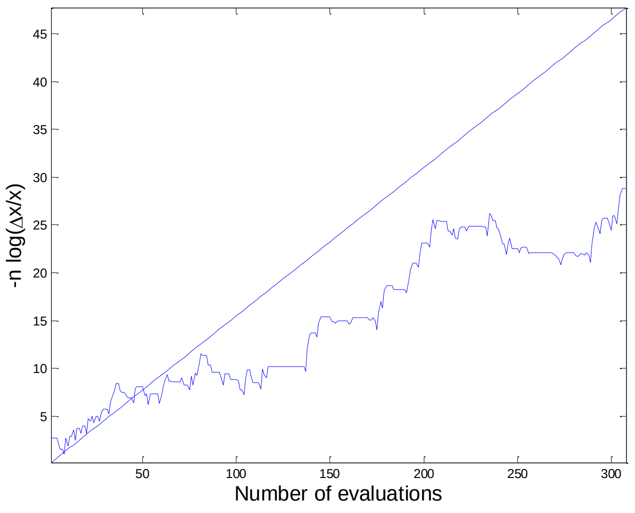

Figure 4. The estimate of information gain for the banana function.

A third test function is the unimodal hump with an added layer of a highly multimodal function, resembling noise.

$$f_3(x_1, x_2) = \sin(\pi x_1)\sin(\pi x_2) + 0.05\sin(500\pi x_1 + \frac{\pi}{2})\sin(500\pi x_2 + \frac{\pi}{2}) \quad (34)$$

where

$$x_i \in [0, 0.7], \quad i = 1, 2$$

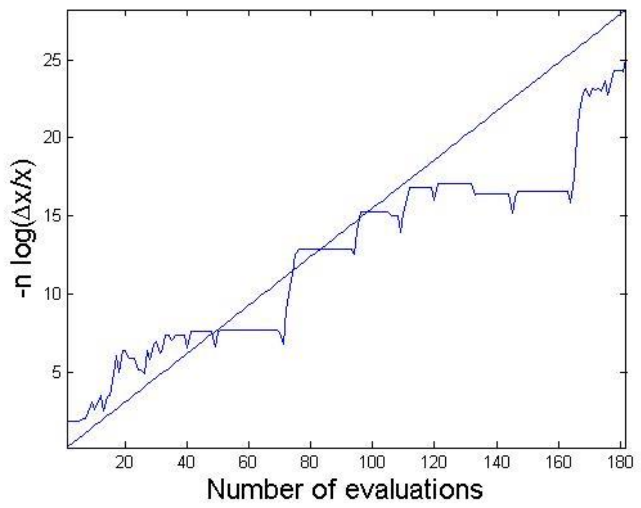

Figure 5. The estimate of information gain for the third test function.

In order to investigate the effect on performance an objective function could be optimized a great many times using the standard Complex method. The success rate of the optimization can then be estimated by dividing the number of optimization runs that come within a certain threshold (e.g. 1%) of the true optimum value by the total number of optimization runs.

In order to take this success rate (or hit rate) into consideration, this can be compared to a target probability of finding the right optimum. Let the hit rate for the optimization be P_{opt} and the target hit rate be P_t. The risk of not finding the optimum is then $(1-P_{opt})$. However, if the optimization is repeated M times, the risk of not finding the optimum in any of these is reduced and by solving the equation

$$\left(1-P_t\right)=\left(1-P_{opt}\right)^M \tag{35}$$

the number of times needed to reduce the risk to $1-P_t$ can be calculated as

$$M=\frac{\log_2\left(1-P_t\right)}{\log_2\left(1-P_{opt}\right)} \tag{36}$$

The performance index in (28) can therefore be modified as

$$\phi^{(1)}=\frac{H_x}{MN_m}=-n\frac{\log_2\varepsilon}{\log_2\left(1-P_t\right)}\frac{\log_2\left(1-P_{opt}\right)}{N_m} \tag{37}$$

Since the $\log_2$ is used (other bases can also be used) this represents one over the number of times it is required to evaluate the objective function in order to get a 50% probability of reaching the optimum. This criterion was presented Krus and Andersson (2003) and also by Schutte et al (2005).

However, for some objective functions it is so simple to find the optimum that the probability of finding the right optimum is P_{opt}=1 (as in the first test function). In this case the criterion moves towards infinity, making it useless at high probabilities of P_{opt}.

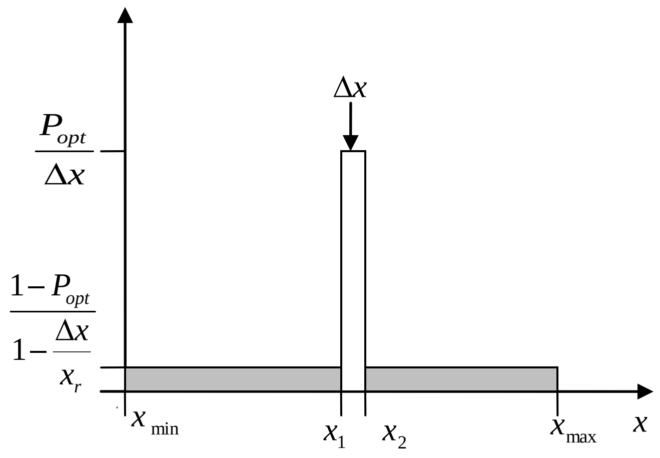
Δx
$\frac{P_{opt}}{\Delta x}$
$\frac{1-P_{opt}}{1-\frac{\Delta x}{x_r}}$
$x_{\min}$
x_1
x_2
$x_{\max}$
x

Figure 6. The probability density function of the found optimum.

It can therefore be useful to turn again to the definition of information entropy to find a better measure of information entropy. First, consider the single variable case. If *x* is a variable that indicates the position of the variable within the design range as in Figure 6, then the probability density function will be

$$\begin{aligned} p(x) &= P_{opt}\frac{1}{\Delta x} : x \in \left[x_{0L}, x_{0H}\right] \\ p(x) &= \frac{(1-P_{opt})}{x_R - \Delta x} : x \in \left[x_{\min}, x_{\max}\right] \setminus \left[x_{0L}, x_{0H}\right] \end{aligned} \tag{38}$$

The information entropy can then be calculated as

$$\begin{aligned} H_x &= \int_{x_{\min}}^{x_{\max}} p(x)\log_2(p(x)x_R)dx = \\ &= \int_{x_{\min}}^{x_{0L}} \frac{(1-P_{opt})}{1-\Delta x / x_R}\log_2\left(\frac{1-P_{opt}}{1-\Delta x / x_r}\right)dx + \int_{x_{0L}}^{x_{0H}} P_{opt}\frac{x_R}{\Delta x}x\log_2\left(P\frac{x_r}{\Delta x}\right)dx + \\ &+ \int_{x_{0H}}^{x_{\max}} \frac{(1-P_{opt})}{1-\Delta x / x_R}\log_2\left(\frac{(1-P_{opt})}{1-\Delta x / x_R}\right)dx = \\ &= (1-P_{opt})\log_2\left(\frac{1-P_{opt}}{1-\Delta x / x_R}\right) + P_{opt}\log_2\left(\frac{P_{opt}}{\Delta x / x_R}\right) \end{aligned} \tag{39}$$

where $x_R = x_{\max} - x_{\min}$. With $\delta_x = \Delta x / (x_{\max} - x_{\min})$ and $\Delta x = x_{0H} - x_{0L}$ the information entropy can then be written

$$H_x = (1-P_{opt})\log_2\left(\frac{1-P_{opt}}{1-\delta_x}\right) + P_{opt}\log_2\left(\frac{P_{opt}}{\delta_x}\right) \tag{40}$$

The general expression for the information entropy for *n* variables is

$$H_x = \int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} p(x_1, \dots x_n)\log_2(p(x_1, \dots x_n))dx_1 \dots dx_n \tag{41}$$

This yields

$$H_x = (1-P_{opt})\log_2\left(\frac{1-P_{opt}}{1-\prod_{i=1}^{n}\delta_{x,i}}\right) + P\log_2\left(\frac{P_{opt}}{\prod_{i=1}^{n}\delta_{x,i}}\right) \tag{42}$$

It can be noted that the second term is usually the dominant one. With ε_x as the cut-off tolerance criterion in x, which is usually a small value, the performance index can be written as

$$\phi^{(2)} = \frac{H_x}{N_m} = \frac{1}{N_m}\left(\left(1-P_{opt}\right)\log_2\left(\frac{1-P_{opt}}{1-\varepsilon_x^n}\right) + P_{opt}\log_2\left(\frac{P_{opt}}{\varepsilon_x^n}\right)\right)$$
$$\left(\approx \frac{1}{N_m}P_{opt}\log_2\left(\frac{P_{opt}}{\varepsilon_x^n}\right)\right) \tag{43}$$

This entropy rate based performance index (ERI) is slightly different from the one represented by (29). For a given (small) value of ε_x the two criteria are very similar up to about P_{opt}=0.7, above which (29) shows a rapid increase towards infinity as P_{opt} approaches 1.

Using the test function in (32), and running one set of one thousand optimization runs with the Complex method, where the stop criterion was set to ε=0.003, a hit rate of P_{opt}=0.67, and an average value of the number of function evaluations N_m=264 are obtained. The performance index in bits/evaluation could then be calculated as

$$\phi^{(2)} = \frac{H_x}{N_m} = \frac{1}{N_m}\left((1-P_{opt})\log_2\left(\frac{1-P_{opt}}{1-\delta_x^n}\right) + P_{opt}\log_2\left(\frac{P_{opt}}{\delta_x^n}\right)\right) =$$
$$\frac{1}{264}\left((1-0.67)\log_2\left(\frac{1-0.67}{1-0.003^2}\right) + 0.67\log_2\left(\frac{0.67}{0.003^2}\right)\right) = 0.039 \tag{44}$$

This reflects the fact the assumption that during the optimization the probability density function changes from a uniform, equal probability in all areas of the design space, into a function with a concentration of probability at the found solution.

There is also a third performance measure that can be used as an alternative to the one based on information theory. This is to compare the performance criterion with a random search. If a random search with no-return is used, the probability of finding the optimum after $N_{eval,r}$ evaluations, can simply be written

$$P_{opt} = N_{eval,r}\delta_x^n \tag{45}$$

This means that the probability of finding the true optimum within the bounds given by δ_x approaches 1 when all locations have been visited within tolerance δ_x. From this expression it is possible to calculate the number of evaluations needed for a given bound and a given probability. This number can then be compared to the actual number of evaluations and probability of success for another method. The reason for this criterion is the "no free lunch theorem" propounded by Wolpert D.H., G. Macready (1997), which states that taken over the set of all objective functions no method is better than the random search. This performance index can then be written

$$\phi^{(3)} = \frac{N_{eval,r}}{N_m} = \frac{P_{opt}}{N_m \delta_x^n} \tag{46}$$

However, for the rest of this study the *entropy rate based performance index* (ERI), from equation (43), is used.

5. Optimization of the Complex-RF method

The next step is to use the Complex-RF method to optimize the optimization parameters in the Complex-RF method itself. In order to obtain a result that gives acceptable performance for different objective functions, a representative set of objective functions where chosen. Of course, if the characteristics of the objective function to be optimized is known, a test function as close as possible to that one should be chosen, or at least an objective function with similar difficultes. In this example, the performance was measured on three objective functions: the unimodal hump (29), the Rosenbrock banana (33), and a unimodal hump with an added low-amplitude high-frequency function that represents noise (37). The unimodal hump was chosen as the performance for simple objective functions should not be compromised to much. Because it is a well-known test function, the Rosenbrock banana was chosen to represent an objective function with some difficulty. Finally, the noisy function was chosen since noise is an effect that sometimes occurs when dynamics simulation models are used to evaluate the objective function.

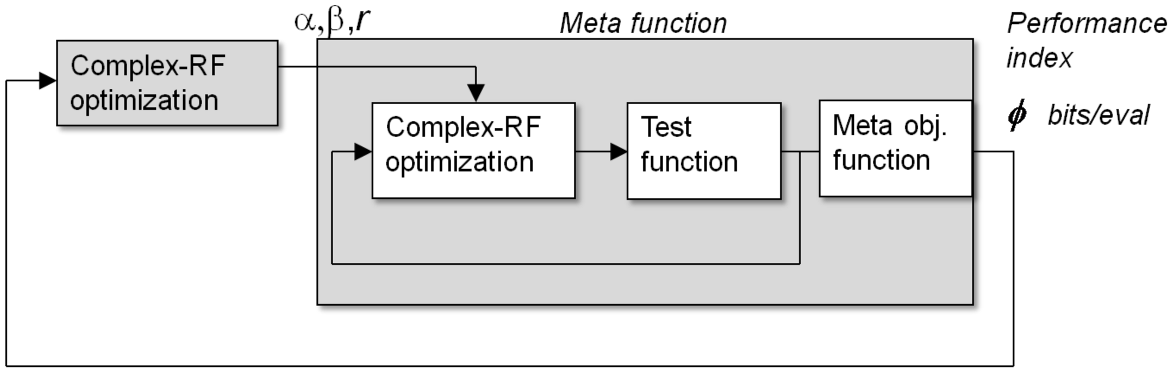

Figure 7. Meta optimization of the Complex-RF algorithm.

Using optimization it is then possible to optimize the reflection factor α , the randomization factor β and the forgetting factor γ . The number of parameter sets in the Complex was set to k=4. First, the Complex-RF is optimized for each test objective function individually. The parameters used for the meta optimization itself were

$$\alpha = 1.4$$
$$\beta = 0.6$$
$$\gamma = 1.0$$

The ERI, for the three different test functions with the individually optimized optimization parameters were

$$\phi_1 = 0.35$$
$$\phi_2 = 0.074$$
$$\phi_3 = 0.045$$

In order to find the best set of optimization parameters for all the objective functions, the meta objective function was set to a combination of the performance indices calculated using eq. (43), for the three test functions. This can be written

$$\phi_{tot} = 1/\left(\left(\frac{\phi_{10}}{\phi_1}\right)^{\kappa} + \left(\frac{\phi_{20}}{\phi_2}\right)^{\kappa} + \left(\frac{\phi_{30}}{\phi_3}\right)^{\kappa}\right)^{1/\kappa} \tag{47}$$

where

$$\kappa = 3$$
$$\phi_{10} = 0.21$$
$$\phi_{20} = 0.07$$
$$\phi_{30} = 0.07$$

In this way low values of the performance criteria will be penalized. The numerators are weights that loosely reflect target values for the performance criteria of the three test functions. These are loosely based on the utopia points but are reduced for the first objective function since it has such a high performance criterion and the third is increased to the same level as the second to further emphasize this. For each parameter set the test equations were optimized 200 times in order to estimate the hit rate and obtain an average of the performance. Using equation (47) as the objective function, an optimization of the optimization yielded the following values; α=1.26, β =0.28 and γ =0.24. The evolution of the parameters is shown in Figure 8.

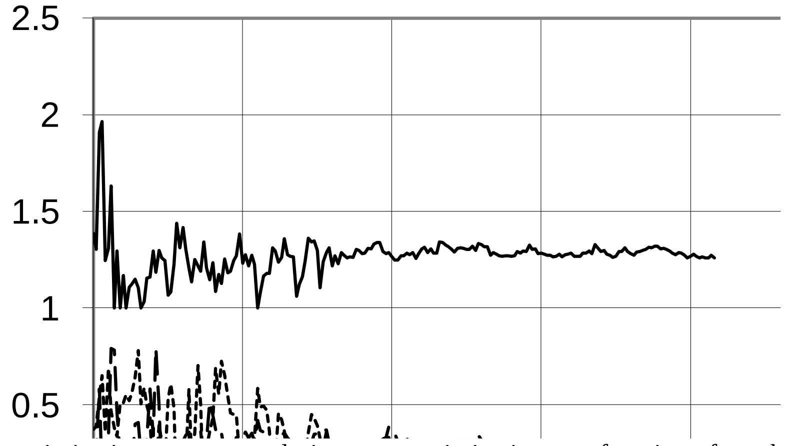

Figure 8. Evolution of the optimization parameters during meta optimization as a function of number of evaluations.

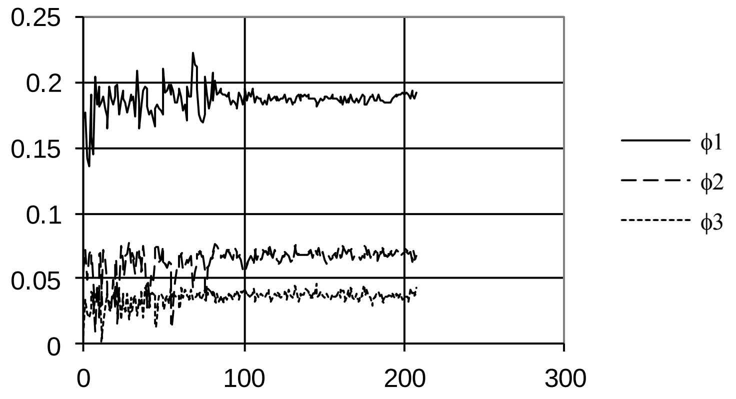

Figure 9. Evolution of the individual performance criterion as a function of number of evaluations.

In order to check the influence of the addition of the β and the γ parameter they were alternatively set to zero as shown in Table 1.

Table 1. The optimized results, showing the ERI bit rate, the number of evaluations and the hit rate for the different objective functions.

	f_1	f_2	f_3	α	β	γ
ϕ	0.192	0.069	0.041	1.26	0.28	0.24
N	87.5	169.1	181.5			
P	1.00	0.73	0.52			

Table 2. The optimized results and the variation if β and γ are omitted.

	$\phi(f_1)$	$\phi(f_2)$	$\phi(f_3)$	α	β	γ
Optimized	0.192	0.069	0.041	1.26	0.28	0.24
	0.190	0.050	0.002	1.26	0	0.24
	0.206	0.066	0.021	1.26	0.28	0
	0.209	0.041	0.0029	1.26	0	0

A sensitivity analysis was also performed by using variations ($\Delta = 0.1$) of the parameter values around the optimum. This yields the following sensitivity matrix.

Table 3. Table of sensitivities.

	α	β	γ
$\phi(f_1)$	-0.116	-0.011	-0.031
$\phi(f_2)$	0.072	-0.055	-0.02
$\phi(f_3)$	0.024	0.128	0.044

or the normalized sensitivities (divided by the performance criteria)

Table 4. Normalized sensitivities.

	α	β	γ
$\phi(f_1)$	-0.603	-0.057	-0.161
$\phi(f_2)$	1.040	-0.795	-0.289
$\phi(f_3)$	0.591	3.153	1.084

Here it can be seen that at the optimum, it is the performance for the last test objective function (f_3) that benefits from an increase in the β and γ values, while the sensitivity is low for these for the unimodal hump (f_1) and the banana function (f_2).

However, with these values a performance index of 0.069 bits/evaluation was achieved for the Rosenbrock's banana, which should be compared with the value for the original version of the algorithm which was 0.041 bits/evaluation. The performance of the unimodal hump is very slightly degraded while the "noisy" function, on the other hand, is improved by a factor of more than ten, as can be seen from Table 2.

6. Objective function temperament

The ERI can also be used to create a kind of difficulty index for the objective function; this is here called the *objective function temperament factor*, OTF for a given optimization method. Due to its wide applicability, robustness and predictive behaviour, the Complex-RF with a suitable parameter set like the one produced here could be a candidate for use as a reference method. The OTF can be defined as the quotient between the ERI of a reference function, $\phi(f_{ref})$, such as a simple unimodal hump (1-modal), divided by the ERI of the objective function, $\phi(f)$, of interest,

$$\Gamma_f^{Complex-RF} = \frac{\phi(f_{ref})}{\phi(f)} \tag{48}$$

For the Rosenbrock's banana function the index is then

$$\Gamma_{f_2}^{Complex-RF} = \frac{\phi(f_{1-\text{modal}})}{\phi(f_2)} = \frac{0.2}{0.069} = 2.90 \tag{49}$$

A more interesting case is perhaps a function with four distinct modes.

$$f_{4\text{mode}} = \sin \pi x_1 \sin \pi x_2 + 3\sin(2.9\times\pi x_1)\sin(2.9\times\pi x_2) \tag{50}$$

where the parameters are in the intervals

$$\begin{aligned} x_1 &\in [0, 0.7] \\ x_2 &\in [0, 0.7] \end{aligned} \tag{51}$$

The resulting OTF is then

$$\Gamma_{4-\text{modal}}^{Complex-RF} = \frac{\phi(f_{1-\text{modal}})}{\phi(f_{4-\text{modal}})} = \frac{0.2}{0.057} = 3.4 \tag{52}$$

which for this case with a uniform distribution of the local optima, is rather close to the modality. As can be expected, the modality greatly influences the difficulty index. The authors believe this is a quantity that combined with experience can be useful in estimating the number of evaluations needed for optimization. Using (28) and (48) the number of evaluations needed for optimization with the precision indicated by H_x,would be

$$N = \frac{H_x}{\phi_f} = \Gamma_f^{method} \frac{H_x}{\phi_{ref}} \tag{53}$$

Note that if N is considered to be analogue to effort or "energy" needed for the optimization and that in thermodynamics, the change in energy can be calculated as:

$$\Delta E = T\Delta S \tag{54}$$

where ΔS is the change in thermodynamic entropy and T the temperature. In chemical reactions temperament is used to indicate a virtual temperature in conversion of chemical energy into electrical energy. Hence the temperament T_o of the optimization process could be defined as:

$$T_o = \frac{1}{\phi_f} \tag{55}$$

Hence Γ_f^{method} would be the *objective function temperament factor*, OTF.

7. Discussion

The Complex-RF method has over the years been applied to a range of optimization problems. In particular it is very suitable for simulation-based optimization. It has been applied to problems ranging from aircraft actuation systems, to industrial robots and hydraulic machinery, where simulation models have been used to calculate the objective function, see for example Krus and Andersson (2003). In this paper it has also been turned towards itself since it was also used to tune its own parameters. Interestingly, the properties that make it a very useful method for optimization of a wide range of problem also make it difficult to optimize precisely, since the optimum is rather flat (hence it is robust towards variation in its parameters), and of course the statistical nature of the method makes the meta objective function noisy even if many test runs are made to evaluate performance at every point. An alternative choice of method (more effective) for the meta optimization might be a response surface approach.

The entropy rate based index (ERI) introduced here, uses the final region of uncertainty for the optimization variables. This is very clear when using a method such as the Complex method. However, in other methods with less pronounced convergence, such as for example Genetic Algorithms (GA) it is less obvious. In these situations the distance of the point closest to the true optimum can be used instead. Sometimes, such as in integer problems, the exact solution might eventually be found. In this situation it might be better to look at the number of function evaluations (and distance to optimum) just before the solution collapses to the exact value. It should also be noted that lack of an inherent method to estimate accuracy of the optimization at a given time is a problem in GA methods, since it also makes it difficult to establish a stop criterion.

Finally, another property, although outside the scope of this paper, that is becoming increasingly relevant, is the ability of a method to utilize multi-core architectures, and a performance criterion needs to be adjusted also for that situation. Here Genetic Algorithms stand out, since a whole population can be evaluated in parallel, as opposed to gradient based methods that are inherently sequential algorithms. However, this property can also be accounted for, if another measure of evaluation cost is used than the number of function evaluations. One straightforward way is to simply measure the time used for an optimization run, which for the GA is proportional to the number of generations.

8. Conclusions

In this paper the Complex-RF method is analyzed and optimized using itself for meta optimization, using a range of representative problems.

In order to evaluate an optimization method a performance index is needed. Here, a single expression, based on information theory, the entropy rate index (ERI), has been derived that produces a balance between computational cost, the probability of finding the optimal solution, and the tolerance of the found optimum, i.e. the parameter convergence. This index can be used both to optimize parameters within the optimization algorithm itself, and to compare different optimization algorithms. Furthermore, based on this, the objective function temperament factor (OTF) is proposed, which can be used to characterize the difficulty of different problems when using a particular optimization algorithm. It is thus possible to state how difficult an objective function is to optimize, compared to a reference objective function.

9. References

Afsar, M.H., 2008., Penalty adapting ant algorithm: application to pipe network optimization. *Journal of Engineering Optimization*, Vol. 40, No. 10, pp.969–987.

Box M. J., 1965 A new method of constrained optimization and a comparison with other methods, *Computer Journal*, Vol. 8, pp. 42-52.

Gavel H., Ölvander J., Krus P., Optimal aircraft fuel system design, AIAA Journal of Aircraft, vol 43., No5, pp. 1334-1340, 2006.

Jaynes, E. T. (1981). Entropy and search theory. *First Maximum Entropy Workshop*. University of Wyoming.

Keane, A. J., 1995. Genetic algorithm optimization of multi-peak problems: studies in convergence and robustness. Artificial Intelligence in Engineering, 9(2), pp.75-83.

Khompatraporn C., Pinter J.D., Zabinsky Z. B., 2005. Comparative assessment of algorithms and software for global optimization, *Journal of global optimization* 31: pp. 613–633, Springer

Krus P., Jansson A., Palmberg J-O. , 1993. Optimization using simulation for aircraft hydraulic system design, *Proceedings of IMECH International Conference on Aircraft Hydraulics and Systems*, London, UK.

Krus P., Gunnarsson S., 1993. Numerical Optimization for Self Tuning Electrohydraulic control systems, *Proceedings of JHPS International Symposium on Fluid Power*, Tokyo, Japan.

Krus P. J Andersson (Ölvander), 2003. Optimizing optimization for design optimization", Proceedings of DETC'03 2003 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference Chicago, Illinois USA.

Kullback, S., and Leibler, R. A., On information and sufficiency, *Annals of Mathematical Statistics* 22: pp.79-86. 1951.

Liao, B. & Luus, R., 2005. Comparison of the Luus–Jaakola optimization procedure and the genetic algorithm. *Engineering Optimization*, 37(4), pp.381-396.

Liu J-L., Chen, C-M., 2009. Improved intelligent genetic algorithm applied to long-endurance airfoil optimization design. *Journal of Engineering Optimization*, Vol. 41, No. 2, pp.137-154.

Luus, R., Sabaliauskas, K. & Harapyn, I., 2006. Handling inequality constraints in direct search optimization. *Engineering Optimization*, 38(4), pp.391-405.

Manetsch, T.J. & Cabrera, A.D., 1991. Use of concurrent processing with the adaptive complex method for global optimization of large dynamc systems, Man and Cybernetics, *IEEE Transactions on. IEEE Transactions on systems, man, and cybernetic*, 21(2), pp.442-445.

Pedersen, M.E.H. & Chipperfield, A.J., 2009. Simplifying particle swarm optimization. Applied Soft Computing, pp.1-32.

Pettersson M., Ölvander J., Drive Train Optimization for Industrial Robots, IEEE Transactions on Robotics, Vol25, No.6, pp.2047-2052, 2009.

Pierce, J. G. (1978). *A New Look at the Relation Between Information Theory and Search Theory*. *Security*. Arlington.

Reklaitis G.V., Ravindran A., Ragsdell K.M., 1983, *Engineering Optimization*, John Wiley & Sons Inc.

Schutte J.F., R.T. Haftka, 2005. Improved Global Convergence Probability Using Independent Swarms, *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*. Austin, Texas, USA,

Shannon D., 1948 A mathematical theory of communication, *Bell Syst. Tech*. J. 27 379.

Rosenbrock, H. H., 1960 An automatic method for finding the greatest or least value of a function, *The Computer Journal* 3: pp.175–184.

Smit, S.K., Eiben, A. E., 2009. Comparing parameter tuning methods for evolutionary algorithms. *2009 IEEE Congress on Evolutionary Computation*, pp.399-406.

Wolpert D.H., Macready, G., 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*. Vol 1, No 1,

Wang, Y., Liu, H., Cai, Z., Zhou, Y., 2007. An orthogonal design based constrained evolutionary optimization algorithm. *Engineering Optimization*, Vol. 39, No. 6, pp.715–736.

Younis, A. & Dong, Z., 2010. Trends, features, and tests of common and recently introduced global optimization methods. *Engineering Optimization*, 42(8), pp.691-718.

Zahara, E., Hu, C-H., 2008. Solving constrained optimization problems with hybrid particle swarm optimization. .*Journal of Engineering Optimization*, Vol. 40, No. 11, pp.1031–1049