Differential Evolution Research – Trends and Open Questions

Rainer Storn

Rohde & Schwarz GmbH & Co KG
P.O. Box 80 14 69
81614 München, Germany
rainer.storn@freenet.de

Summary. Differential Evolution (DE), a vector population based stochastic optimization method has been introduced to the public in 1995. During the last 10 years research on and with DE has reached an impressive state, yet there are still many open questions, and new application areas are emerging. This chapter introduces some of the current trends in DE-research and touches upon the problems that are still waiting to be solved.

1 Introduction

It has been more than ten years since Differential Evolution (DE) was introduced by Ken Price and Rainer Storn in a series of papers that followed in quick succession [1, 2, 3, 4, 5] and by means of an Internet page [6]. DE is a population-based stochastic method for global optimization. Throughout this chapter the term optimization shall always be equated with minimization without loss of generality. The original version of DE can be defined by the following constituents.

1) The population

$$\begin{aligned} &\mathrm{P}_{\mathbf{x},g} = \left(\mathbf{x}_{i,g}\right), \quad i = 0,1,...,Np-1, \quad g = 0,1,...,g_{\max}, \\ &\mathbf{x}_{i,g} = \left(x_{j,i,g}\right), \; j = 0,1,...,D-1. \end{aligned} \tag{1}$$

where Np denotes the number of population vectors, g defines the generation counter, and D the dimensionality, i.e. the number of parameters.

2) The initialization of the population via

$$x_{j,i,0} = \mathrm{rand}_j[0,1) \cdot \left(b_{j,\mathrm{U}} - b_{j,\mathrm{L}}\right) + b_{j,\mathrm{L}}. \tag{2}$$

The D-dimensional initialization vectors, $\mathbf{b}_{\mathrm{L}}$ and $\mathbf{b}_{\mathrm{U}}$ indicate the lower and upper bounds of the parameter vectors $\mathbf{x}_{\mathrm{i,j}}$. The random number generator, $\mathrm{rand}_j[0,1)$, returns a uniformly distributed random number from within the range [0,1), i.e., $0 \le \mathrm{rand}_j[0,1) < 1$. The subscript, j, indicates that a new random value is generated for each parameter.

U.K. Chakraborty (Ed.): Advances in Differential Evolution, SCI 143, pp. 1–31, 2008.
springerlink.com

3) The perturbation of a base vector $\mathbf{y}_{i,g}$ by using a difference vector based mutation

$$\mathbf{v}_{i,g} = \mathbf{y}_{i,g} + F \cdot \left(\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}\right). \tag{3}$$

to generate a mutation vector vi,g. The difference vector indices, r1 and r2, are randomly selected once per base vector. Setting yi,g = xr0,g defines what is often called classic DE where the base vector is also a randomly chosen population vector. The random indexes r0, r1, and r2 should be mutually exclusive. There are also variants of perturbations which are different to Eq. (3) and some of them will be described later. For example, setting the base vector to the current best vector or a linear combination of various vectors is also popular. Employing more than one difference vector for mutation has also been tried but has never gained a lot of popularity so far.

4) Diversity enhancement

The classic variant of diversity enhancement is crossover [1, 2, 3, 4, 5, 6, 7] which mixes parameters of the mutation vector $\mathbf{v}_{i,g}$ and the so-called *target vector* $\mathbf{x}_{i,g}$ in order to generate the *trial vector* $\mathbf{u}_{i,g}$. The most common form of crossover is uniform and is defined as

$$\mathbf{u}_{i,g} = u_{j,i,g} = \begin{cases} v_{j,i,g} & \text{if } \left(\text{rand}_j[0,1) \le Cr\right) \\ x_{j,i,g} & \text{otherwise.} \end{cases} \tag{4}$$

In order to prevent the case $\mathbf{u}_{i,g} = \mathbf{x}_{i,g}$ at least one component is taken from the mutation vector $\mathbf{v}_{i,g}$, a detail that is not expressed in Eq. (4). Other variants of crossover are described by Price, Storn and Lampinen [7].

5) Selection

DE uses simple one-to-one survivor selection where the trial vector ui,g competes against the target vector xi,g. The vector with the lowest objective function value survives into the next generation g+1.

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g} & \text{if } f\left(\mathbf{u}_{i,g}\right) \le f\left(\mathbf{x}_{i,g}\right) \\ \mathbf{x}_{i,g} & \text{otherwise.} \end{cases} \tag{5}$$

Please note that the presentation as well as notation has been chosen slightly different from the original papers[1, 2, 3, 4, 5]. Along with the DE algorithm came a notation [5] to classify the various DE-variants. The notation is defined by DE/x/y/z where x denotes the base vector, y denotes the number of difference vectors used, and z representing the crossover method. For example, DE/rand/1/bin is the shorthand notation for Eq. (1) through Eq. (5) with yi,g = xr0,g. DE/best/1/bin is the same except for yi,g = xbest,g. In this case xbest,g represents the vector with the lowest objective function value evaluated so far. With today's extensions of DE the shorthand notation DE/x/y/z is not sufficient any more, but a more appropriate notation has not been defined yet.

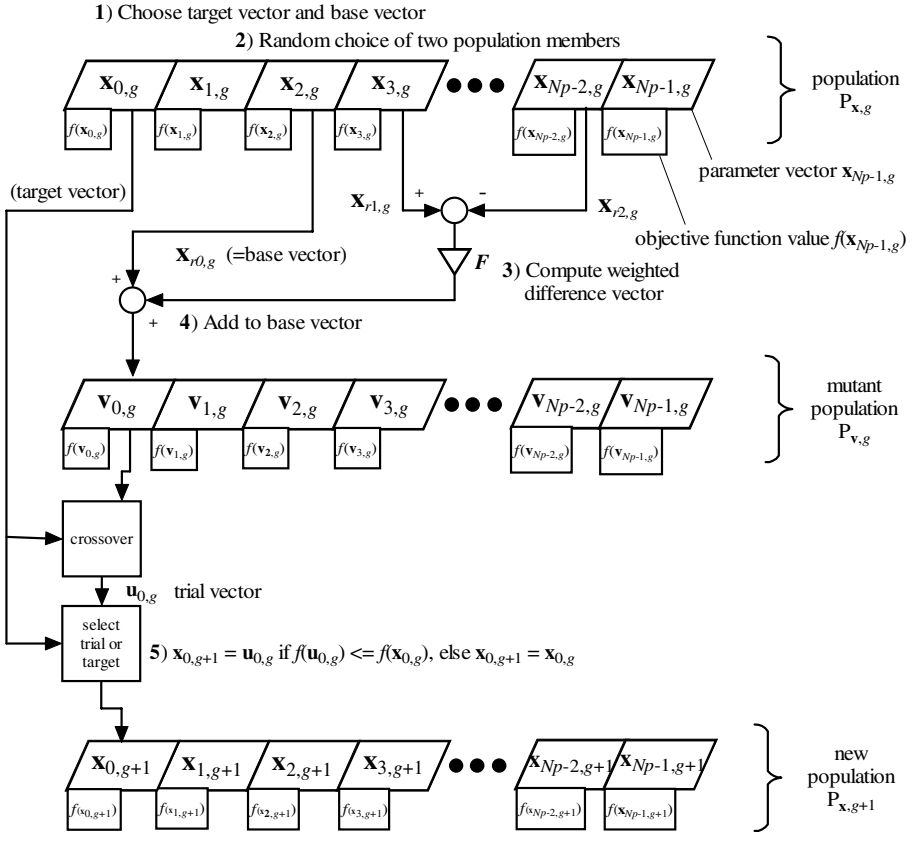

Fig. 1. Flowchart of classical DE [7]

In order to be able to represent DE pictorially the flowgraph representation in Figure 1 was contrived because flowgraphs are very common in the engineering world. The DE-flowgraph representation seemed to be ideal to convey DE's simplicity and first appeared in the DE-article published in the Dr. Dobb's journal at 1997 [4], a magazine for computer programmers. The article spawned a lot of interest for DE among practitioners which Kenneth Price and Rainer Storn concluded from the large number of e-mails they received in which DE was attributed very good convergence along with simplicity. Simplicity is an asset which is very important to anyone who considers optimization to be a necessary but not the primary task. DE's simplicity allowed many practicing engineers and researchers from very diverse disciplines to use global optimization without the need to be an optimization expert.

Interestingly enough, DE received attention only very slowly from fellow researchers in the evolutionary computation community, even though it performed very well on the first international contest on evolutionary computation in Nagoya as early as 1996 [2]. DE's lack of attention might have been due to a lack of understanding concerning its inner workings. More light was shed upon these in 2002

when Daniela Zaharie published a beautiful article [8] that enlightened the convergence of DE from theoretical point of view. Also the contour matching properties of DE, a phrase coined by Kenneth Price, were not explicitly advocated until 2005 [7] and only vaguely described as the self-steering property of DE.

For the following discussions the terms *population vectors* and *population points* will be used interchangeably, depending on the circumstances. Talking about vectors is usually more appropriate when issues concerning the parameters of the vectors or the vector arithmetic used to generate new vectors are elaborated. Speaking in terms of points, however, is usually more convenient when the discussion concentrates on the sampling of the objective function surface. It should be kept in mind that the points are simply the endpoints of the vectors.

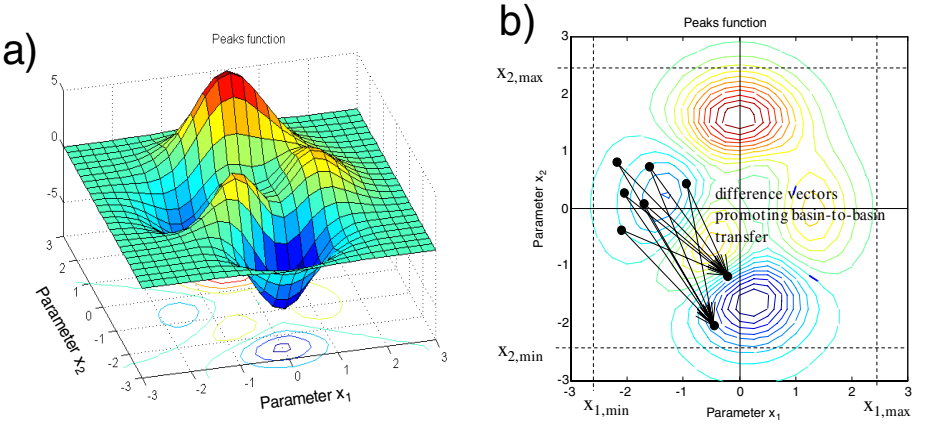

Fig. 2. Peaks function a) and illustration of difference vectors b) that promote transfer of points between two basins of attraction of the objective function surface

Some explanations concerning contour matching are in order. Contour matching means that the vector population adapts such that promising regions of the objective function surface are investigated automatically once they are detected. To this end an important ingredient besides selection is the promotion of *basin-to-basin* transfer where search points may move from one basin of attraction, i.e. a local minimum, to another one. Figure 2 illustrates that DE in fact supports basin-to-basin transfer by yielding a certain amount of difference vectors that are able to generate new trial points in the lower basin of attraction when the base points stem from the upper left basin of attraction.

Professor Jouni Lampinen from Lappeenranta University of Technology, Finland, [7] was one of the first scientists who was intrigued by DE's potential and not only did a lot of seminal DE-research but also started to maintain a bibliography of DE-related papers [9]. A look into this bibliography on the Internet reveals that its maintenance has been halted after 2002. The reason for this stop was that the number of papers began to increase at such a large rate after the year 2002 that it was impossible to keep the bibliography up-to-date and complete.

A simple search for "Differential Evolution" using any kind of search engine on the Internet supports the above statement and shows that DE-research is in full swing.

There are basically four main directions of DE-research that can be identified:

- Basic DE research
  Here the inner workings and theoretical aspects of DE are investigated. Objective functions involved are usually unconstrained. The goal is to better understand DE, identify its weaknesses and improve it in an overall fashion. It is worth mentioning that the majority of the research in this category is empirical. Purely theoretical treatments are rare as it is the case for EAs in general. This is probably due to the situation that those scenarios which lend themselves to feasible theoretical investigations rarely represent the complexity of real-world problems.
- Problem Domain Specific research
  In this area the problem formulation and how DE can be adapted to it is under scrutiny. For example, constraints, time variations, and number of objectives of an objective function are of importance but also effects of dimensionality and parameter granularity, i.e. discreteness are considered.
- Application Specific research
  This research domain is similar to the problem domain specific case, however, certain applications can be much more specific than the general problem domain they belong to. For example, the traveling salesman problem belongs to the problem domain of combinatorial optimization, but its specifics narrow down the heuristics one may use in order to solve the problem.
- Computing Environment related research
  In the real world computational efficiency of DE is often crucial to make design problems tractable. Some problems call for parallel computations while others have to deal with limited memory or processing power.

In the following selected areas from the research domains mentioned above will be discussed. I have tried to highlight those that still exhibit many open questions and hence constitute rewarding research topics, but I am aware that I did not provide a complete picture. For example, the vast domain of multi-objective optimization using DE has not been covered at all, but it will be treated in another part of the book.

2 Basic DE Research

In the early days DE was only marginally understood concerning its strengths and weaknesses. By twisting and tuning the various constituents of DE, i.e. initialization, mutation, diversity enhancement, and selection of DE as well as the choice of the control variables it was tried to make it a foolproof and fast optimization method for any kind of objective function, even though the *No Free Lunch Theorem* (NFL) by Wolpert and Macready (1997) [10] suggested already that such a panacea could not exist. Nevertheless, many real-world problems seem to be of the kind that they are very well amenable to be treated by DE. And even though there will be no cure-all-optimization for every problem, DE can nevertheless be improved also in a general sense.

2.1 The Control Variables Np, F, and Cr

Trying to tune the three main control variables Np, F, and Cr and finding bounds for their values has been a topic of intensive research [7, 8, 11, 12, 13, 14, 15, 16]. An important result was presented by Daniela Zaharie [8] where she proved that the mutation scale factor F should never be smaller than F_{crit} where

$$F_{\text{crit}} = \sqrt{\frac{\left(1 - Cr/2\right)}{Np}} \,. \tag{6}$$

Another important result from Price, Storn, Lampinen [7] was that only high values of Cr guarantee the contour matching properties of DE. In addition, only when Cr=1 is the mean number of function evaluations for an objective function and its rotated counterpart the same, i.e. in this case DE is called *rotationally invariant*. This does not mean, however, that low values of Cr should always be avoided. Low values of Cr are advantageous for separable functions, since the search concentrates on the axes of the coordinate system as outlined in [7]. The rule of thumb values for the control variables given by Storn and Price [5]:

1. $F \in [0.5, 1.0]$
2. $Cr \in [0.8, 1.0]$
3. $Np = 10 \cdot D$

are valid for many practical purposes but still lack generality. Gämperle [11] reported that the control variable settings for F, Cr, and Np can be quite difficult to find, and some objective functions are sensitive to the proper setting. This finding was also stated by Liu and Lampinen [12]. Therefore research trends go towards finding the best settings of F, Cr, and Np automatically [13, 14, 15, 16, 17]. One recent approach by Brest *et al.* [17] uses F and Cr as additional parameters to evolve for each population vector, an idea pioneered by Schwefel [18]. Hence each parameter vector has D + 2 parameters with the last two parameters containing an individual F and Cr for the particular vector. If the trial vector wins in the selection process either both F and Cr from the base vector are transported into the winner vector or the individual F and Cr are randomly determined. It is claimed that the most appropriate values for F and Cr will survive in the long run. The results on a reasonably-sized testbed show that the scheme yields improved objective function values after a fixed set of function evaluations, compared to classical DE with F=0.5 and Cr=0.9, and compared to some other DE-variants. However, it is unclear whether the scheme would maintain it superiority if not a fixed number of evaluations but a fixed value-to-reach (VTR) would have been chosen as a goal. It may also be that the encouraging results are due to the occasionally occurring random selection of F and Cr. This kind of randomness known as dither [7] has been found to be advantageous as will be elaborated later. Furthermore the question remains whether the surviving F and Cr gear the optimization towards fast and therefore possibly premature convergence. Hence the area of automatic control parameter determination remains very interesting and a fruitful area of research.

2.2 Perturbation

Perturbation of the base vector by mutation has been treated very early and has lead to various variants of DE such as the one belonging to classical DE/rand/1/bin

$$\mathbf{v}_{i,g} = \mathbf{x}_{r0,g} + F \cdot \left(\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}\right), \tag{7}$$

the mutation being used in DE/best/1/bin

$$\mathbf{v}_{i,g} = \mathbf{x}_{best,g} + F \cdot \left(\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}\right), \tag{8}$$

the mutation for DE/current-to-best/1/bin

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F \cdot \left(\mathbf{x}_{best,g} - \mathbf{x}_{i,g}\right) + F \cdot \left(\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}\right). \tag{9}$$

and the variant for DE/best/2/bin

$$\mathbf{v}_{i,g} = \mathbf{x}_{best,g} + F \cdot \left(\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g} + \mathbf{x}_{r3,g} - \mathbf{x}_{r4,g}\right). \tag{10}$$

In fact many more linear combinations of vectors may be used for mutation, a generalization of which can be written as

$$\mathbf{v}_{i,g} = \mathbf{y}_{i,g} + F \cdot \frac{1}{N} \sum_{n=0}^{N-1} \left(\mathbf{x}_{r(2n+1),g} - \mathbf{x}_{r(2n+2),g}\right). \tag{11}$$

with $\mathbf{y}_{i,g}$ being the base vector. The base vector should be distinct from the other vectors in Eq. (11). Most commonly used are the mutation schemes represented by Eq. (7) and Eq. (8) with the latter being more greedy. Recently Price and Rönkkönen [19] investigated Eq. (11) for the case $\mathbf{y}_{i,g} = \mathbf{x}_{i,g}$ and N=0. In [7] the effect of recombination as a perturbation method

$$\mathbf{v}_{i,g} = \mathbf{y}_{i,g} + F_{rec} \cdot \left(\frac{1}{N} \sum_{n=0}^{N-1} \left(\mathbf{x}_{r(2n+1),g} + \mathbf{x}_{r(2n+2),g}\right) - \mathbf{y}_{i,g} \right). \tag{12}$$

for the case $\mathbf{y}_{i,g} = \mathbf{x}_{i,g}$ and N=0 has been elaborated. Eq. (12) is a generalization of the Nelder and Mead reflection operation [20] and defines a point between $\mathbf{y}_{i,g}$ and the centroid of the vectors used in the recombination sum. So far no single perturbation method has turned out to be best for all problems which, of course, doesn't come as a surprise with regard to the NFL [10]. Nevertheless all the various methods need further investigation under which circumstances they perform well. In practice this information can be very important because it may save many computations or may even be crucial for the solution of a certain problem.

2.3 Diversity Enhancement

One of the most fundamental aspects of mutation-based DE is the fact that vector perturbations are generated from the Np·(Np-1) nonzero difference vectors of the population rather than employing a predetermined probability density function. This leads to one of the main assets of DE: *contour matching* [7]. The contour matching

property can be observed in Figure 3 through Figure 5 which show the DE-population and the difference vector distribution for Np=8 on the peaks function the latter of which is defined by

$$f(x_1,x_2)=3(1-x_1)^2\cdot\exp\left(x_1^2+(x_2+1)^2\right)-10\left(\frac{x_1}{5}-x_1^3-x_2^5\right)\cdot \\ \cdot\exp\left(x_1^2+x_2^2\right)-\frac{1}{3}\cdot\exp\left((x_1+1)^2+x_2^2\right). \tag{13}$$

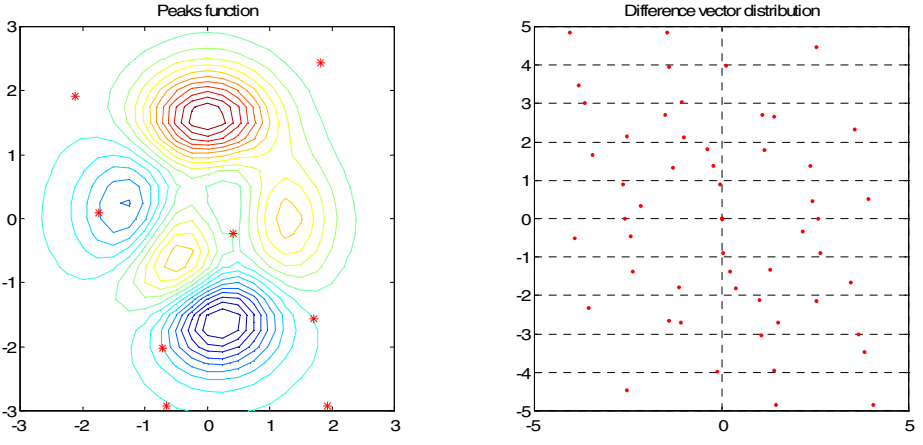

Fig. 3. Generation g=1 using Np = 8

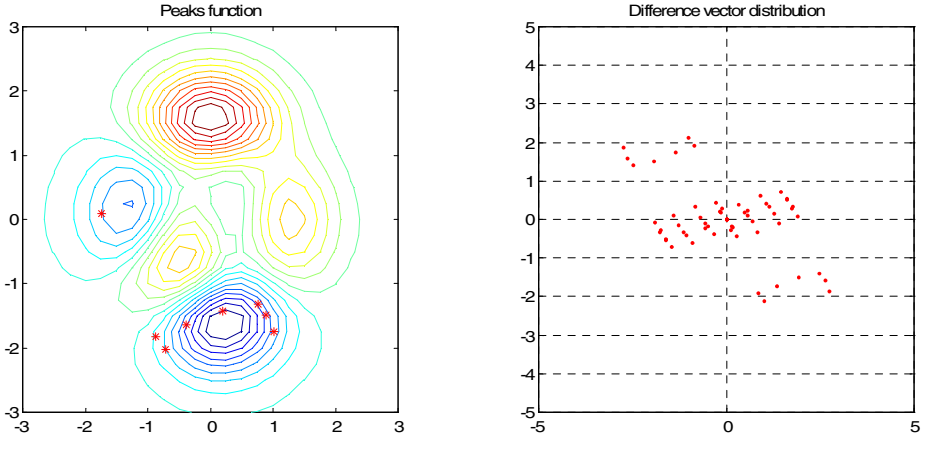

Fig. 4. Generation g=10 using Np = 8. The difference vector distribution (only endpoints shown) exhibits three main clouds where the outer ones promote the transfer between two basins of attraction.

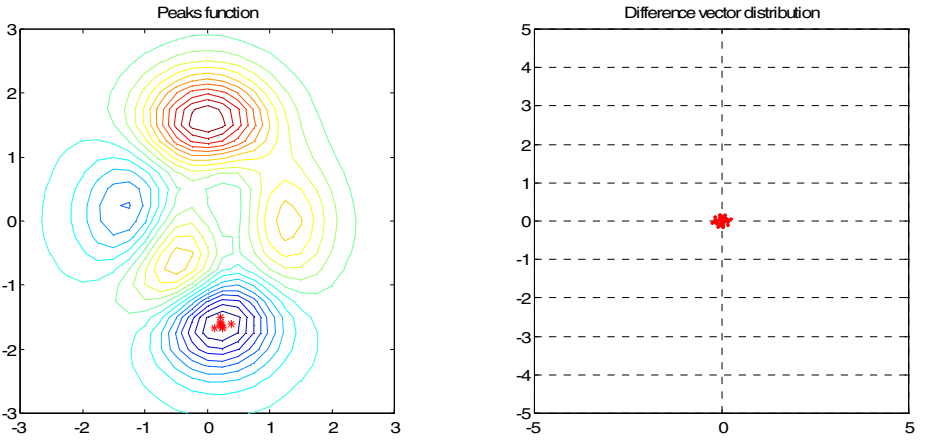

Fig. 5. Generation g=20 using Np = 8. Now the difference vector distribution fosters the local search of the minimum the vector population is enclosing.

It is intriguing to see the difference vector distribution adapt to the landscape of the objective function. This self-adaptivity renders DE's mutation based perturbation superior to mere Gaussian- or Cauchy-based types in most cases. However, Figure 3 through Figure 5 also reveals a weakness:

In the endeavour to obtain fast convergence the population size Np is usually kept low. Due to the limitation of Np·(Np-1) potential perturbation possibilities for a base vector there is a limited possibility to find regions of improvement and hence stagnation [21] can be the price to pay for the low number of Np. The blessing of contour matching may then turn out to be a curse when the contour of the objective function is deceiving and the "matching" leads away from the global optimum.

In order to increase the number of potential points to be searched while still maintaining a low number of Np gives rise to the various strategies for diversity enhancement, certainly one of the most interesting and rewarding areas of DE research today. The basic idea is simply to find some hopefully contour-matching and rotationally invariant way to generate more potential points without increasing the number Np of population members. As has been mentioned above one method for diversity enhancement has always been a part of DE, crossover.

Crossover

Crossover, i.e. mixing parameters of the target and the mutant vector in order to get the trial vector (see Eq. (4)) has been introduced to DE from the beginning [1, 2, 3, 4, 5, 6]. It was felt that mutation alone is too restrictive as a perturbation method, just as genetic algorithms [22] require some random mutation in addition to the dominant recombination mechanism to make the optimization work properly. So both DE and genetic algorithms have some dominant method of change plus an additional ingredient which slightly breaks up the mechanics of the dominant perturbation. Figure 6, however, forebodes that crossover has the potential to destroy the directional information provided by the difference vectors for the sake of increasing diversity. In fact it has been shown [7] that DE's contour matching property is lost

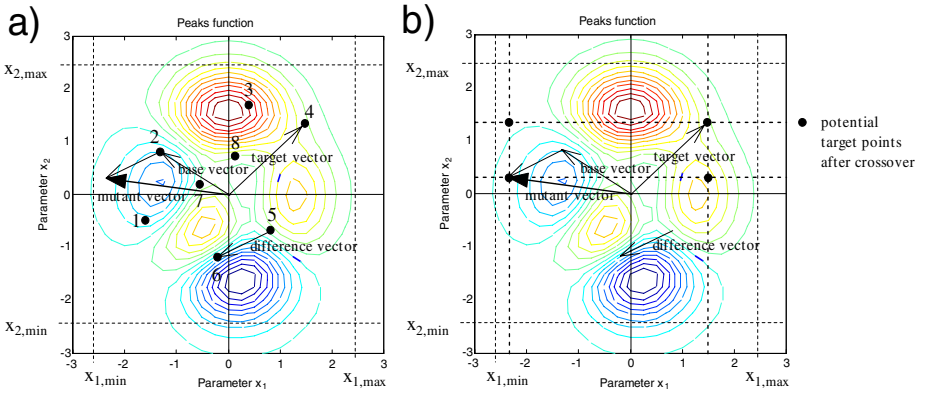

Fig. 6. Example for a population of Np=8 points and a mutation step a). The figure on the right b) shows the potential points when using crossover.

when strong crossover is used (e.g. Cr=0.1) and that in this case DE has a strong tendency to search along the main parameter axes, a property that is in fact beneficial for separable objective functions. Yet for real-world applications separability is rarely present. Parameter dependency seems to be the rule rather than the exception instead.

Another deficiency of crossover is that it is not rotationally invariant, i.e. optimization results obtained for a certain objective function do not directly translate to the rotated counterpart of this function. The differences in potential crossover points for two coordinate systems with the same origin are depicted in Figure 7. Despite its deficiencies the DE-literature reveals that crossover is almost always used. The diversity enhancing features of crossover seem to outweigh its disadvantages, at

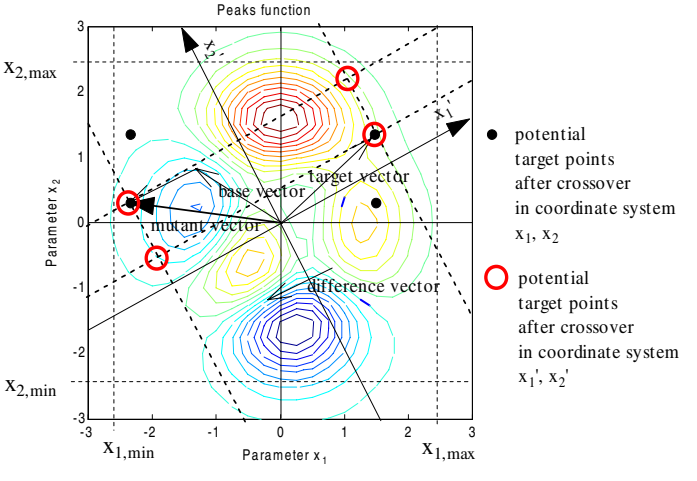

Fig. 7. Potential trial points after crossover for coordinate system x_1, x_2 and system x_1', x_2'

least if used lightly, i.e. if Cr stays close to 1 for the case of non-separable objective functions.

Dither

The term *dither* has been defined in [7] and presumably has been used by early practitioners of DE. The first reported publication advocating its use, however, seems to have been launched by Karaboga and Ökdem [23] in a Turkish Journal even though the term "dither" had not been employed. In [23] the scale factor F was randomized according to

$$F_{dither} = F_l + rand_g(0,1) \cdot \left(F_h - F_l\right) \tag{14}$$

for every generation g. Independently of [23] Das, Konar, and Chakraborty [24] have reported improvements in DE's convergence when using dither. In [24] dither had been applied to every difference vector i=0, 1, …, Np-1 rather than on a generational basis.

$$F_{dither} = F_l + rand_i(0,1) \cdot \left(F_h - F_l\right) \tag{15}$$

More variants are conceivable, for example changing F using some randomization different from uniform. A pictorial representation of dither is provided in Figure 8.

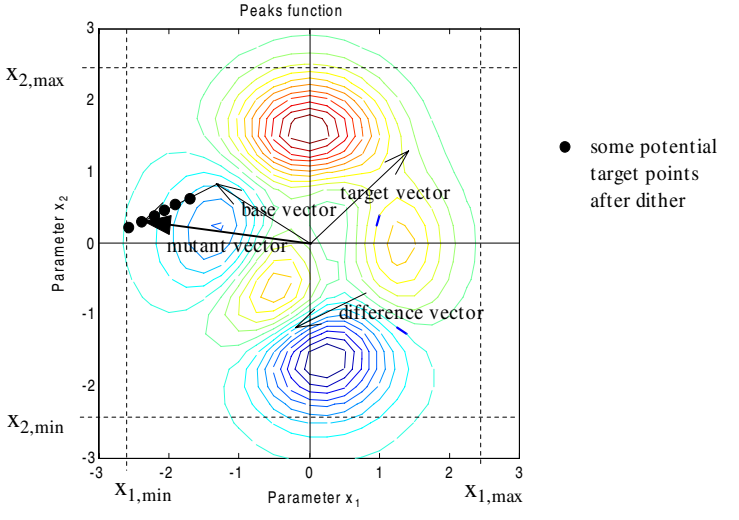

Fig. 8. Pictorial representation of dither which simply randomizes the mutation scale factor F and hence does not compromise DE's contour matching

Besides from improving DE's convergence behavior on time-independent objective functions dither also improves DE's handling of noisy objective functions [25]. Since dither is rotationally invariant and preserves the contour matching property this diversity enhancing method should always be used.

Jitter

Jitter as defined in [7] is somewhat similar to dither in that the scaling factor F is randomized. However, F is randomized for each single parameter j=0, 1, …, D-1 and for every new mutant vector i according to

$$F_{jitter,i} = F \cdot \left(1 + \delta \cdot \left(rand_j[0,1) - 0.5\right)\right) \tag{16}$$

Jitter has not been treated a lot in the literature. Zaharie [8] has used a Gaussian randomized form of jitter for the theoretical convergence proof of DE. Storn [26] has implemented it in a commercial program for digital filter design, and also Lampinen [27] reportedly has used but never published it. For jitter it seems to be very important that δ be small, e.g., δ=0.001. In fact δ may even be randomized itself. Figure 9 visualizes jitter and shows the effect of randomizing all parameter directions. The effect is a square cloud of potential points centered at the tip of the mutant vector.

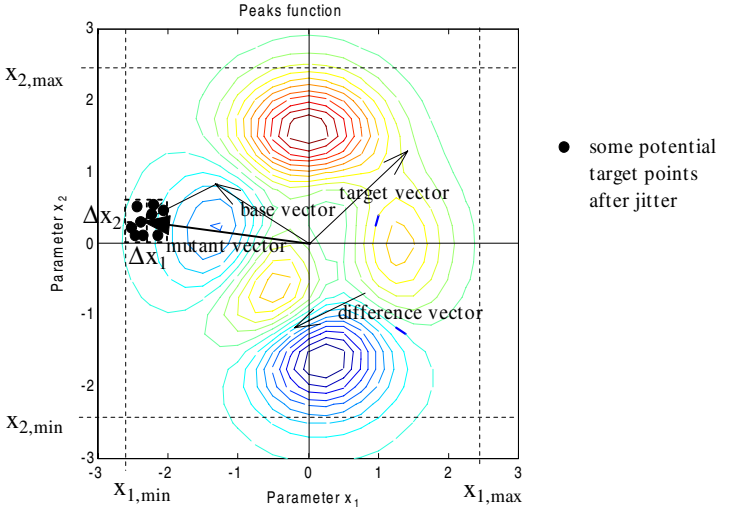

Fig. 9. Jitter randomizes the difference vector in all parameter directions

As stated in [7] jitter is not rotationally invariant, but for small δ this deficiency is negligible. The benefit of jitter seems to be that Np can be reduced so that convergence is sped up without loosing robustness, but jitter-research is still in its infancy. Empirics indicate that jitter works well for non-deceiving objective functions. In this context non-deceiving shall mean that if an objective function posesses a strong global gradient information then it also leads towards the vicinity of the global minimum. For example, Corana's paraboloid [28] which is riddled with small local minima would be non-deceiving.

In addition it also seems to be beneficial to combine jitter with dither [26] as in

$$F_{jitter\&dither,i,g} = \left(F_l + rand_g(0,1) \cdot \left(F_h - F_l\right)\right) \cdot \left(1 + \delta \cdot \left(rand_j[0,1) - 0.5\right)\right), \tag{17}$$

but more research is required with regard to jitter to get conclusive results.

Mixing perturbation techniques

Mixing perturbation techniques is another diversity generating technique that has received some attention in the past. One example is the *Either-Or-Algorithm* proposed by Kenneth Price and described in [7]. This technique counteracts stagnation by choosing at random which perturbation method to use, mutation like in Eq. (11) or recombination like in Eq. (12). Figure 10 provides an example for the differences in potential target points.

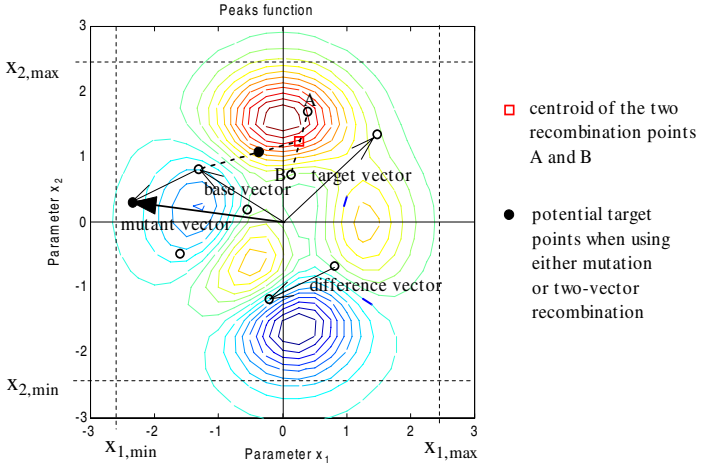

Fig. 10. Potential target points after applying either Eq. (11) or Eq. (12) for N=0

The advantage of this technique is that both rotational invariance and the basin-to-basin transfer property (i.e. contour matching) are preserved. Extensive empirical tests have shown that this approach can be more robust than classical DE while exhibiting a slow-down in convergence. Yet not all varieties of perturbation have been explored to a sufficient extent, e.g. multi-vector mutation or recombination, as well as usage of dither and jitter in addition to mutation and recombination. So there are quite a number of loose ends which need to be investigated further.

Opposition-based points

Another interesting concept for diversity enhancement has been introduced by Rahnamayan, Tizhoosh, and Salama [29] which uses either the mutant vector obtained in the usual way or its opposing point, depending on some probabilistic descision. The scheme is dubbed ODE for *opposition-based DE*. The opposing vector is defined as

$$\mathbf{v}_{i,g,opposed} = \mathbf{x}_{g,\min} + \mathbf{x}_{g,\max} - \mathbf{v}_{i,g}. \tag{18}$$

where $\mathbf{x}_{g,\min}$ and $\mathbf{x}_{g,\max}$ define the momentary extremes for each parameter taken over the entire population at a certain generation g. For the initial generation g=0 the absolute bounds are taken for these extremes. It is interesting to see that the opposing points generation scheme neither fulfills rotational invariance nor has the capability

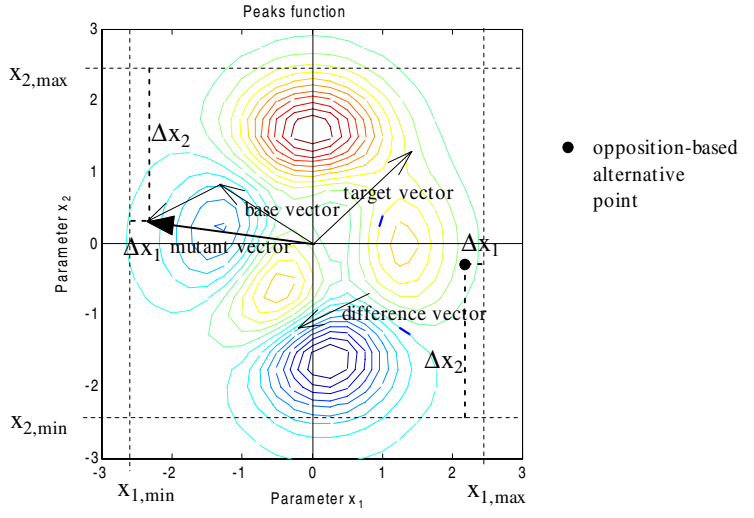

Fig. 11. Illustration of the construction of an opposition-based population point. Note that this point generating scheme does neither satisfy rotational invariance nor basin-to-basin transfer.

for basin to basin transfer. Figure 11 provides an example for opposition based vector generation.

Rahnamayan, Tizhoosh, and Salama [29] report a convergence speed up compared to classical DE. The reason for this may stem from two properties of the scheme:

1. The opposing points are not chosen on an individual basis but an entire population is generated with some probability JR (in [29] with JR=0.3) on a generational basis. Once this population of opposing points is generated there are 2·Np points available, Np from the current population and another Np from the population of points which is opposite to the current population. Out of these 2·Np points the Np best ones are chosen to form the next generation of points. In the evolutionary programming community this selection scheme is called *elitist*, or (μ+λ)-selection [7]. As indicated above this elitist selection is not used for every generation but only with a probability JR. The authors of [29] refer to this scheme as generation jumping. Elitist schemes usually speed up convergence because only the best points are retained. On the downside the chances for premature convergence are increased. Generation jumping might offer a good balance between elitist and one-to-one survivor selection.
2. The occasional generation jumping breaks up the vector generation scheme of DE just like the probabilistically occurring mutation breaks up the crossover scheme for GAs. This may hinder contour matching once in a while, but on the other hand it increases diversity, and the (μ+λ)-selection counteracts the loss of focus towards the optimum by being more greedy than the selection scheme of DE.

Again, the reasons for ODE's success and potential deficiencies need to be investigated further. Also a combination with other diversity enhancing strategies is worth investigating.

The strategies above are not the only diversity enhancing strategies that can be found in the literature. For example, in Ali [30] an extra distribution, the so-called β-distribution is applied to enhance the diversity of DE. Enhancing diversity is certainly a very interesting area of DE-research and hopefully more fruitful ideas in that domain will appear.

2.4 Controlling the Vector Population

One-array vs. Two-array

Classic DE uses two populations in order to allow computation on parallel computers or processors, something which is becoming increasingly important especially since the advent of multicore processors [80]. But in fact the very first algorithm that Kenneth Price came up with used just one single vector population array. This simplified version has been described in [7] in the light of saving memory on limited resource devices. In Feoktistov [31] this scheme is investigated further and extended to transversal DE where an individual may undergo several mutation/evaluation steps before it is compared to the target vector. Unfortunately in [31] the consequences, benefits or drawbacks are not regarded for a sufficiently large test set, so more research is needed to evaluate this idea. Transversal DE bears some similarity with hybrid DE versions that employ gradient algorithms or other greedy techniques for local search (see chapter 4.1.) in that the trial vector undergoes several improvement steps before it is compared to the target vector.

Selection Methods

Selection methods have been extensively discussed in [7]. The main methods of interest are:

1. Elitist $(\mu+\lambda)$-selection where the best μ individuals out of $\mu+\lambda$ individuals are selected. For DE usually $\mu=\lambda=Np$ is used.
2. Tournament selection with one-to-one survivor selection as in classic DE.

There have not been too many investigations on alternatives to DE's one-to-one selection, but it can be said that elitist selection is accelerating convergence while making the optimization more prone to premature convergence. In addition, elitist selection makes parallel computation more difficult. An in-depth numerical comparison of selection methods using an extensive testbed, however, is still lacking.

3 Problem Domain Specific Research

So far we have regarded DE as an optimization method to minimize objective functions without specifying the makeup of these objective functions. The tacit assumption when looking at "basic research" of evolutionary optimization is generally that objective functions to be minimized have a single global minimum, are

potentially multimodal and nonlinear, and have a moderate number of parameters. Real-life problems, however, are often more complicated than that. Optimizations may include bounds constraints, inequality constraints, equality constraints, or they may even consist of constraints only without any objective to be optimized. The latter problem is well known as constraint satisfaction problem. If only equality constraints prevail and no objective is present we are encountering a system of equations. So far we have concentrated on parameters stemming from the continuous space, i.e. the floating point domain. Yet problems may also include discrete parameters or consist of discrete parameters only. If there are only discrete parameters and these parameters follow no metric, which means that there is no smaller, equal, or greater relationship, then we are looking at a combinatorial problem in the strict sense [7]. There are even more dimensions to optimization problems which the spider diagram in Figure 12 attempts to visualize. Many of these problem domains have been treated in [7] but still a lot of open questions remain. An entire chapter can be written easily for each case depicted in Figure 12 so only a few problem domain types will be sketched in the following in order to illustrate the research potential.

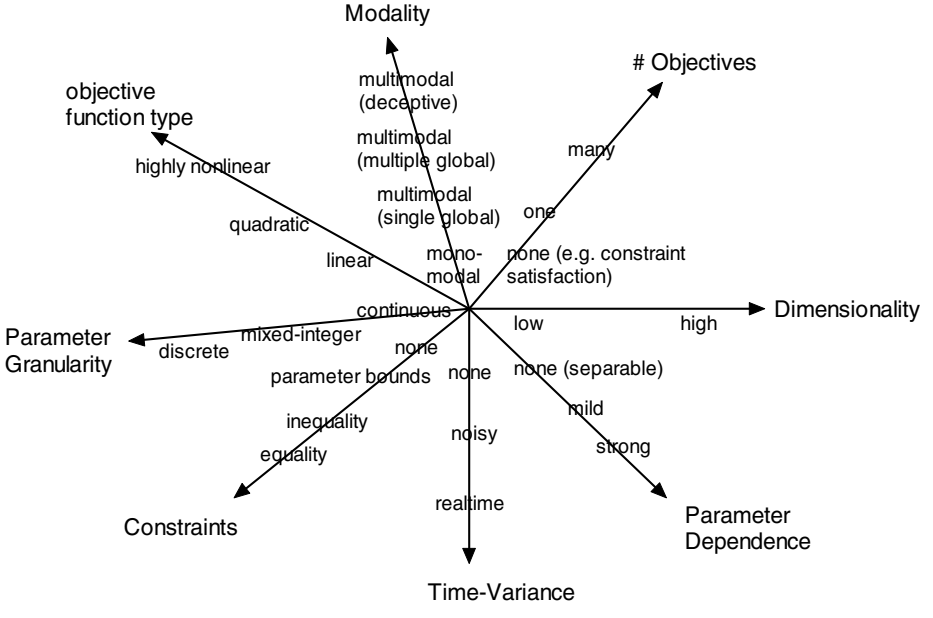

Fig. 12. Spider diagram sketching the various problem domain characteristics

3.1 Objective Functions with Single Objective

Figure 13 shows a classification of objective functions with a single objective regarding three problem domain dimensions, time variance, constraints, and parameter granularity. When available in the literature real-world example applications are provided. Figure 13 illustrates that at present DE is mostly used for time-invariant problems.

Objective Function → Time Variance ↓ Constraints / ↓ Granularity		Time-invariant	Noisy	Time-variant (realtime)
Unconstrained	Continuous	Function approximation [38, 39], Channel Capacity Maximization [48]	Parameter identification [40], Object detection [45], Multisensor Fusion [7]	Acoustic Echo Cancellation [41] (quadratic problem, not solved with DE)
	Mixed integer	Circuit optimization with partially discretized components [49]		
	Discrete			
Constrained	Continuous	Electronic Circuit Design [33, 34, 35, 36], Optimization of Si-H Clusters [43]	Design Centering of Manufacturing Processes [7, 37]	
	Mixed integer	Mechanical Design Optimization [32], Compressor Supply Optimization [44]		
	Discrete	Design of Digital Filters in Signal Processing [7, 26]		
	Combinatorial	Traveling Salesman Problem [7], Cryptography applications [46, 47]		Routing Algorithms in Network Communication, not solved with DE [42]

Fig. 13. Classification of objective function types and some example applications. Problem types where DE-research is either in its infancy or not applicable are indicated with solid-frame table entries.

3.2 Combinatorial Problems

Combinatorial problems in the strict sense are problems where the parameters are discrete, the number of discrete states is finite, and the parameters are not associated with a metric, i.e. one cannot tell which of two different values for the same parameter is greater than the other unless an artificial metric is applied. For example the letters A and F of the alphabet may have an order in the alphabet, but one cannot truly say that A is greater than F or the other way round. Many of the well-known combinatorial problems are also highly constrained like the *traveling salesman problem* (TSP) which we will look at for illustration purposes later on. Even though DE has a good reputation of solving discrete or mixed-integer problems [7, 26, 32, 44, 50, 51, 52] there is no good evidence so far that DE is applicable to strict-sense combinatorial problems, at least if they are heavily constrained. In [7] the topic of combinatorial problems has been discussed, and the success of DE-based solutions to combinatorial problems was attributed to well-chosen repair mechanisms in the algorithm rather than DE-mutation. However, the applicability of DE to strict-sense combinatorial problems is neither proven nor disproven and depends also on finding a discrete operator that corresponds to the difference vector in the continuous domain. In addition it is required that the combination of a base vector and a difference vector (or recombination vector) yields a new valid vector. The validity of the newly generated vector is a big problem for most of the classical combinatorial problems like the TSP.

The traveling salesman problem (TSP)
Let us regard the traveling salesman problem (TSP) as an example to see how DE may be used to solve it and what the difficulties are. The TSP is a universal strict-sense combinatorial problem, and many other strict-sense combinatorial problems can be transformed into a TSP formulation [53]. Hence many findings about DE's performance on the TSP can be extrapolated to other strict-sense combinatorial problems.

Let there be M cities c_m, m=1,2, ..., M. Each city c_m has a distance $d_{m,n} = d_{n,m}$ to some other city c_n, n not equal to m, associated with it. The task in the TSP is to find a graph where all cities are visited and where the total distance

$$D = \sum_{m=1}^{M} d_{m,n} \quad with \quad n \neq m \quad and \quad mutually \quad different \tag{19}$$

is minimized. Figure 14 shows an example of a 5-city tour.

An approach using distances as parameters
In order to apply DE we first have to find an appropriate problem formulation. A natural approach would be to set up the problem vector x which contains all M distances $d_{m,n}$ as parameters, because the arithmetic difference of distances has a meaning and hence is suited for DE. For each city c_i there are M-1 distances $d_{m,n}$ to the other cities. For a finite set of parameters it is helpful to have these parameters in ascending order. So for our five city TSP example the list of distances would be as shown in Table 1. If we look at the first row as well as Figure 14, we see that city c_4 is closest to city c_1. The next closest city is c_2, then c_3, and then c_5. The other rows can be checked in a similar way. Since we know that DE prefers to have continuous parameters we take a table-based approach for non-uniform quantization. In this case

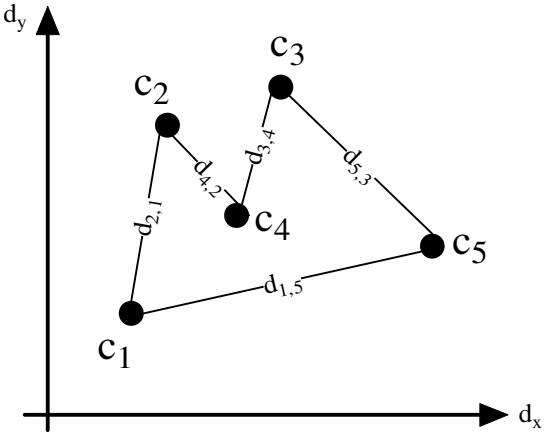

Fig. 14. Example for a city tour in a TSP for five cities [7]

DE's parameters are not the distances themselves but the appropriate array indices. In order to be able to use a continuous parameter vector x for DE we choose

$$\mathbf{x} = \left(x_1, x_2, x_3, x_4, x_5\right)^T \tag{20}$$

with x_m representing a reference to the distance between city c_m and its successor along the travel path. The values x_m are continuous and from the range [-0.5,3.5], so the index of the distances is computed via

$$index_m = floor\left(x_m + 0.5\right) \tag{21}$$

Table 1. Distance table containing the distances from each city c_m to each other city in ascending order

index $_m$ / city c_m	0	1	2	3
c_1	$d_{1,4}$	$d_{1,2}$	$d_{1,3}$	$d_{1,5}$
c_2	$d_{2,3}$	$d_{2,4}$	$d_{2,1}$	$d_{2,5}$
c_3	$d_{3,2}$	$d_{3,4}$	$d_{3,5}$	$d_{3,1}$
c_4	$d_{4,2}$	$d_{4,3}$	$d_{4,1}$	$d_{4,5}$
c_5	$d_{5,4}$	$d_{5,3}$	$d_{5,2}$	$d_{5,1}$

According to Eq. (21) abd the range information for x_m the $index_m$ can assume the values 0, 1, 2, or 3.

While trying to construct a valid path we easily see the constraints. For the first parameter x_1, which represents the distance associated with city c_1, we may choose a value from all the available table entries in Table 1, e.g. $d_{1,3}$. Since each city may be visited only once we must go to city c_3 in the next step because in the first step a path from city 1 to city 3 was selected. In order to prevent traveling back to city 1 the distance

Table 2. If $d_{1,3}$ is selected for city c_1 then $d_{3,1}$ may not be used for city c_3. This is indicated by greyshading.

index $_n$ / city c_m	0	1	2	3
c_1	$d_{1,4}$	$d_{1,2}$	**$d_{1,3}$**	$d_{1,5}$
c_2	$d_{2,3}$	$d_{2,4}$	$d_{2,1}$	$d_{2,5}$
c_3	$d_{3,2}$	$d_{3,4}$	$d_{3,5}$	$d_{3,1}$
c_4	$d_{4,2}$	$d_{4,3}$	$d_{4,1}$	$d_{4,5}$
c_5	$d_{5,4}$	$d_{5,3}$	$d_{5,2}$	$d_{5,1}$

Table 3. If $d_{3,4}$ is chosen the next city to work on is c_4. Now only two distances are free to choose from.

index $_n$ / city c_m	0	1	2	3
c_1	$d_{1,4}$	$d_{1,2}$	**$d_{1,3}$**	$d_{1,5}$
c_2	$d_{2,3}$	$d_{2,4}$	$d_{2,1}$	$d_{2,5}$
c_3	$d_{3,2}$	**$d_{3,4}$**	$d_{3,5}$	$d_{3,1}$
c_4	$d_{4,2}$	$d_{4,3}$	$d_{4,1}$	$d_{4,5}$
c_5	$d_{5,4}$	$d_{5,3}$	$d_{5,2}$	$d_{5,1}$

$d_{3,1}$ in the row for c_3 is excluded from the allowed list as indicated in Table 2 by greyshading.

So our search range has been restricted. Let's assume that we will choose $d_{3,4}$ in the next step, then our next city to consider will be c_4 and the associated parameter is x_4.

For x_4 the choice of available distances is even more restricted, as indicated in Table 3. So all the cities which have already been considered are excluded from the further search. The constraints get tighter and tighter until just one city is left. The last city must be connected to the first one in order to complete the tour.

Now the problem DE faces here becomes evident: Not only is the choice of allowable indexes for each city restricted, but also is this restriction dependent on which city we start the tour with and in which direction we go first. There is another fundamental problem with this approach: DE in general relies on the fact that a small difference vector means that the two parameter vectors are close together. This means that two identical solutions should yield the vector difference zero. However, we can immediately verify that the vectors $\mathbf{x}_a = (d_{1,2} \;\; d_{2,4} \;\; d_{4,3} \;\; d_{3,5} \;\; d_{5,1})$ and $\mathbf{x}_b = (d_{1,5} \;\; d_{5,3} \;\; d_{3,4} \;\; d_{4,2} \;\; d_{2,1})$ describe exactly the same tour but do not yield the vector difference zero. Hence one of DE's biggest assets, the self-adaptivity of the vector difference distribution is severely disturbed because a converged population still might exhibit large difference vectors.

Additional problems arise due to the heavy constraints inherent in the TSP. For example, even if we have a population of valid vectors, the weighted difference of two vectors added to a third one rarely yields a valid tour. If we want to repair this vector we can do this in many ways so that finally not much of DE's working principles are left. In the above case the constraints are dependent on the selection of

the city, so the problem is not invariant. There have been attempts to treat such kinds of problems by not caring about valid solutions at first and simply applying validity as a hard constraint on the vectors [50]. The problem with this approach, however, is that most of the generated vectors are invalid and hence the optimization is very prone to stagnation. There are a number other approaches to optimize the TSP with DE [7], but none of them is really convincing. So successful optimization of heavily constrained strict-sense combinatorial problems using DE still remains to be fairly uncharted territory and leaves substantial room for improvement.

3.3 Design Centering

The problem of *design centering* using DE [37] has been largely left untouched even though this problem is of great importance to manufacturing. The idea is very simple: the design of any technical system usually has to meet certain specifications. Due to imprecisions in the manufacturing process the actual properties of the system often deviate from the nominal ones. The goal of design centering is to estimate the manufacturing imprecisions and to consider them at design time so that the probability that the eventual design violates any of the specifications is minimized.

In mathematical terms the design centering problem can be described as

$$\int_{\mathrm{ROA}} \mathrm{PDF}(\mathbf{x}_0) d\mathbf{x}_0 = \text{maximum}. \tag{22}$$

which means that the D-dimensional parameter vector $\mathbf{x}_0$ should be located such in the so-called *region of acceptability* (ROA) that the deviations of the parameters from their nominal values, which are described by a D-dimensional probability density function, mostly fall into the ROA rather than outside. The ROA is the permitted region in the parameter space within which any actual parameter vector may lie in order to fulfill the design specifications.

As an example Figure 15 shows a nominal magnitude function of a switched-capacitor lowpass filter and also its real-world counterpart after manufacturing. The manufacturing process introduces so-called parasitic capacitors which make the magnitude function violate the tolerance scheme. Figure 16 on the other hand shows the resulting magnitude function after manufacturing if the nominal design has undergone a design centering optimization, i.e. the parasitics are included into the design and the nominal parameter vector is placed within the design center of the ROA. Because the magnitude function lies well between the boundaries of the tolerance scheme it is intuitive that there is some headroom for the parameter values.

In [7] the problem of design centering and its solution via DE has been treated to some extent, and it was suggested that, provided that all population vectors are equally distributed within the ROA, a rough estimate of the design center is the point that maximizes the *center index*:

$$c_j = \sum_{i=1}^{Np} \exp(-d_{ij}), \quad j = 1,2,...,Np, \quad d_{ij} = \left| \mathbf{x}_i - \mathbf{x}_j \right|. \tag{23}$$

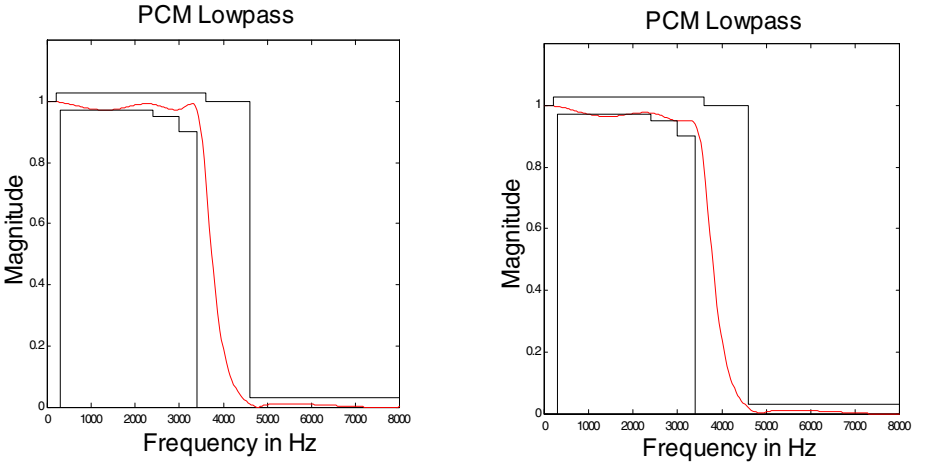

Fig. 15. Lowpass transfer function by standard filter design (left side) and after inclusion of parasitics (right side) [37]

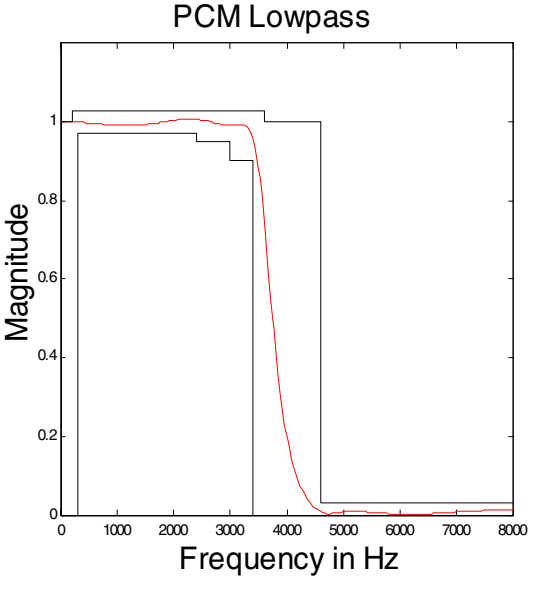

Fig. 16. Transfer function with parasitics but after design centering [37]

The center c_j index increases with the number of vectors that are close to vector $\mathbf{x}_j$. In other words, if a vector has a lot of neighbors then it is probably located fairly close to the design center. For vectors close to the rim of the ROA there is simply not as much space to accommodate many neighbors. Although this claim may appear intuitive, it has not been verified for a large enough testbed, and in fact the usage of DE for design centering problems still offers a lot of research opportunities.

3.4 Time-Variant Objective Functions

All population based optimization routines are mainly geared towards "offline optimization" or, in other words, optimization of time-independent objective functions. Especially when it comes to realtime applications where the minimum-finding process has to occur in a matter of seconds or even milliseconds, like in echo cancellation [41] or routing [42], minimization approaches have to be used that are more greedy and faster converging. Often the minimization problem is simplified to be a quadratic one, so fast gradient methods can be used [41]. Other applications may not necessarily require the global optimum because it will be gone anyway in a matter of seconds [42]. Nevertheless there are problem domains where population based optimizers may be applied. These domains comprise noisy and/or slowly varying multimodal objective functions.

Noisy objective functions
Noisy objective functions frequently occur in practice and are most often due to measurement imprecisions. As an example we may use the parameter identification problem for an induction motor [40]. First certain characteristic voltages of the motor are measured over time and/or over frequency. Then the parameters of a mathematical model of the motor are to be determined such that the absolute difference between the voltages generated by the model and the voltages from the measurements is minimized. The objective function to be minimized is noisy because the voltage measurements are noisy.

It has been reported by Krink, Filipic, Fogel, and Thomsen [54], that classic DE exhibits convergence problems on some noisy objective functions, at least when resampling is used as a noise-mitigating strategy. The idea of resampling is simply to evaluate the same candidate solution m times and to estimate the 'true' fitness value by the mean of the samples [54]. In [54] the question was raised whether thresholding rather than resampling may be a solution. The idea in thresholding [55] is to use a new selection operator for ES, such that a new candidate solution can only replace an existing one if the fitness difference is larger than a threshold τ. A disadvantage, however, is that with τ a new control variable enters the optimization scheme. A hint to the potential solution was already given in [54] when it was noticed that a specific Evolutionary Algorithm, which was compared to DE and showed better performance, used a Gaussian mutation operator. Eventually Chakraborty [24] showed that indeed DE is superior to this particular Evolutionary Algorithm if dither is added to classical DE. Rahnamayan, Tizhoosh, and Salama, [56] provided additional results showing the DE's performance on noisy objective functions can be improved if the evaluation of opposition-based points is added to classic DE resulting in what the authors call ODE. Both methods, dither and ODE, are diversity enhancement methods, so the question arises if there are more diversity enhancement methods which are beneficial for noisy objective function minimization.

Slowly time-variant objective functions
The application of DE to slowly varying objective functions is a very young area of research that has been touched briefly in [7] and which has been more intensely investigated by Mendes and Mohais [57]. The investigation in [57] revolves around

the *moving peaks benchmark* (MPB) and suggests a DE-variant called *DynDE* (Dynamic DE) to approach this. The main ingredients in DynDE are:

1. Usage of several populations in parallel
2. Usage of uniform dither for $F \in [0,1]$ as well as $Cr \in [0,1]$
3. To maintain diversity of the populations two approaches may be chosen:
    a. Reinitialization of a population if the best individual of a population gets too close to the best individual of another population. The population with the absolute best individual is kept while the other one is reinitialized. This way the various populations are kept from merging.
    b. Randomization of one or more population vectors by adding a random deviation to the vector components. Various schemes of randomization are suggested.

The authors conclude that DynDE yields reasonable results, but admit that more research is required to improve this particular DE-variant.

The added dimension of time to the optimization problem requires to deal with many objective function instances at different points in time which makes research very expensive in terms of computational effort. This may be the reason why this problem domain has not been covered to a greater extent so far.

Since there is only little available literature there remains a lot of room to further explore the very interesting problem area of slowly varying objective functions.

4 Application Specific Research and Consequences for DE

It has been mentioned before that specific applications may bear some properties that make it worthwhile revisiting or extending DE so that the optimization matches the problem in the best possible way. Generally should any knowledge about the problem be incorporated into the optimization method and/or the objective function in order to make it more efficient. In the following we will look at the problem of digital filter design to illuminate this.

4.1 An Example: Digital Filter Design

Digital Filter design is a field of signal processing where specialized numerical methods govern the field [61]. To perform a filter design in practice is generally a matter of seconds, once the correct specifications are available. In some cases, though, there may be applications that require unconventional designs which cannot be performed using the standard methods [7, 26]. This is where DE can be of help, but it must be kept in mind that the filter designer is used to short design times which is why DE's convergence should be fast. It appears that the objective functions involved in digital filter design are non-deceiving, albeit multimodal. To illustrate this claim we look at the design task where a magnitude function $A(\Omega)$ has to fit into a tolerance scheme. An example for such a scheme is provided in Figure 17 which represents a so-called bandpass filter because only signal-portions with the normalized frequencies

Ω from the passband remain more or less unattenuated while the other portions get suppressed to a large degree. The equations needed to define A(Ω) are:

$$H(z)=\frac{U(z)}{D(z)}=\frac{\sum_{n=0}^{N_z} a(n)\cdot z^{-n}}{1+\sum_{m=1}^{M_p} b(m)\cdot z^{-m}}=A_0\frac{\prod_{n=0}^{N_z-1}\left(z-z_0(n)\right)}{\prod_{m=0}^{M_p-1}\left(z-z_p(m)\right)} \tag{24}$$

with

$$z=e^{j2\pi\Omega}=\cos(2\pi\Omega)+j\cdot\sin(2\pi\Omega),\qquad j=\sqrt{-1}\,. \tag{25}$$

From Eq. (25) it is evident that for Ω=0 there must be z=1 and for Ω=0.5 there must be z=ı. Finally we define

$$\mathrm{A}(\Omega)=\left|\mathrm{H}\left(\mathrm{e}^{\mathrm{j}2\pi\cdot\Omega}\right)\right|=\sqrt{\mathrm{Re}\left(\mathrm{H}\left(\mathrm{e}^{\mathrm{j}2\pi\cdot\Omega}\right)\right)^2+\mathrm{Im}\left(\mathrm{H}\left(\mathrm{e}^{\mathrm{j}2\pi\cdot\Omega}\right)\right)^2} \tag{26}$$

From Eq. (25) and Figure 17 it becomes clear that a range of Ω ∈ [0, 0.5] transforms itself into the upper semi-circle of the z-plane. It is typical for a digital filter that the poles zp(m) are located in the passband while the zeros z0(m) are located in the stopband. So already by defining the tolerance scheme the approximate locations of poles and zeros are known. In [26] the fact is utilized that if the parameters of the objective function are not the coefficients a(n) and b(m) but the zeros z0(n) and the poles zp(m) then applying jitter together with dither in the DE-variant DE/best/1/bin works extremely well. Reasonable values for the control variables are Cr=0.95 and Np = 2·D, …, 5·D. The mutation method used in [26] is described by Eq. (27)

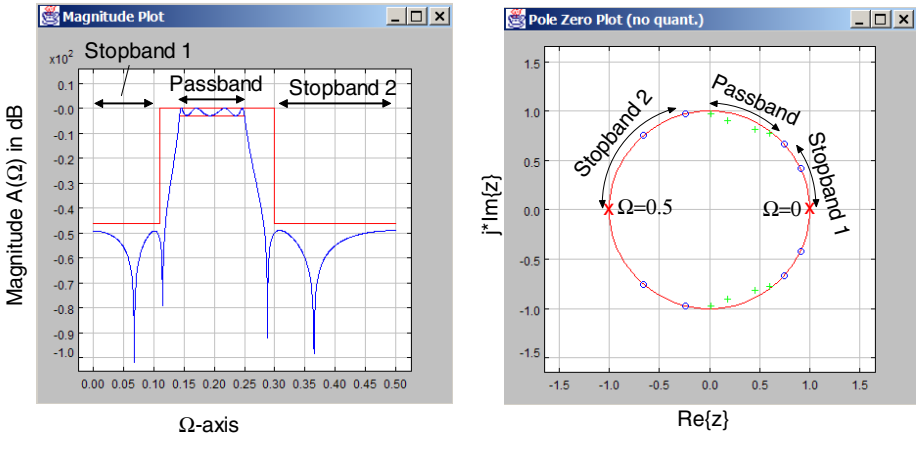

Fig. 17. Tolerance scheme in red and magnitude A(Ω) in blue (left figure) and the corresponding poles (+) and zeros (o) in the z-plane (right figure)

$$u_{j,i,g} = \begin{cases} v_{j,i,g} = x_{j,best,g} + (F_{dither,g} - 0.001 \cdot rand_j^{(1)}[0,1)) \cdot \left(x_{j,r1,g} - x_{j,r2,g}\right) & \text{if} \quad rand_j^{(2)}[0,1) \le Cr \\ x_{j,i,g} \quad \text{otherwise} \end{cases}$$
$$with \quad F_{dither,g} = 0.5 + rand_g[0,1) \cdot 0.5 \tag{27}$$

The superscripts k on the random number randj(k)[0,1), k=1,2, in Eq. (27) shall denote that the numbers for k=1 and for k=2 are generated independently. Empirical evidence has shown that the positive effect of dither is small while jitter together with low values of Np is considerably speeding up convergence. These results come somewhat unexpected compared to the findings reported in [7] that jitter combined with DE/best/1/bin does not always perform well. The success of the described DE variant for this specific application is probably attributable to the benign nature of the objective functions, but more evidence needs to be gathered to corroborate this assumption.

What can be learned from this example is the advice to not consider the findings about DE-variants obtained from a large testbed as being universally applicable. This is also in line with the NFL [10]. For a specific application it may be worthwhile to revisit certain variants of DE depending on the properties of the objective functions at hand.

There are other potential possibilities to accelerate the convergence of DE-based digital filter design, one of which is using hybrid methods. In [62, 63, 64, 65] hybrid methods have already been used successfully. The basic idea usually is to refine one or more points from the DE-population by applying a fast-converging local search method like the Nelder and Mead optimization [20], dynamic hill climbing [66], or gradient type of algorithms [60, 67]. This refinement may take place for every DE-generation or after a certain amount of DE-generations. Gradient algorithms, however, are probably not appropriate since A(Ω) is not always differentiable.

5 More Topics and Outlook

The topics mentioned in the sections above are only some of many. Quite a few important topics have just been touched upon or not been discussed at all in order not to extend the chapter beyond a reasonable size. A few more important DE research topics are:

- DE for multiple objectives and multiple constraints [68, 69, 70, 71, 72, 73]
- DE for multiple global minima [58, 59, 60]
- Stopping criteria [74]
- Hybrid versions [62, 63, 64, 65, 66]
- DE for various computational environments [7, 31, 75, 76, 77, 78, 79]

The remaining chapters of this book will shed more light on many interesting topics of DE-research but are certainly unable to present a solution to all the questions raised in this chapter. So optimization with the help of DE remains a challenging and interesting research area for many years to come.

References

[1] Storn, R., Price, K.V.: Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces (1995) Technical Report TR-95-012, ICSI (March 1995), ftp://ftp.icsi.berkeley.edu/pub/techreports/1995/tr-95-012.ps.Z

[2] Storn, R., Price, K.V.: Minimizing the real functions of the ICEC 1996 contest by differential evolution. In: Proceedings of the 1996 IEEE international conference on evolutionary computation, Nagoya, Japan, pp. 842–844. IEEE Press, New York (1996)

[3] Storn, R.: On the usage of differential evolution for function optimization. In: Smith, M.H., Lee, M.A., Keller, J., Yen, J. (eds.) Proceedings of the 1996 biennial conference of the North American fuzzy information processing society – NAFIPS, Berkeley, CA, USA, June 19–22, pp. 519–523. IEEE Press, New York (1996)

[4] Price, K., Storn, R.: Differential evolution: a simple evolution strategy for fast optimization. Dr. Dobb's Journal 22, 18–24 (1997)

[5] Storn, R., Price, K.V.: Differential Evolution – a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11(4), 341–359 (1997)

[6] Storn, R.: Homepage of DE (2002), http://www.icsi.berkeley.edu/~storn/code.html

[7] Price, K., Storn, R., Lampinen, J.: Differential Evolution – A Practical Approach to Global Optimization. Springer, Heidelberg (2005)

[8] Zaharie, D.: Critical values for the control parameters of differential evolution algorithms. In: Matoušek, R., Ošmera, P. (eds.) Proceedings of MENDEL 2002, 8th international conference on soft computing, Brno, Czech Republic. Brno University of Technology, Faculty of Mechanical Engineering, June 5–7, pp. 62–67. Institute of Automation and Computer Science, Brno (2002)

[9] Lampinen, J.: A bibliography of differential evolution algorithms. Technical report, Lappeenranta University of Technology, Department of Information Technology, Laboratory of Information Processing(October 16, 1999), http://www.lut.fi/~jlampine/debiblio.htm

[10] Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE transactions on evolutionary computation 1(1), 67–82 (1997)

[11] Gämperle, R., Müller, S.D., Koumoutsakos, P.: A Parameter Study for Differential Evolution. In: Grmela, A., Mastorakis, N.E. (eds.) Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, pp. 293–298. WSEAS Press (2002)

[12] Liu, J., Lampinen, J.: On setting the control parameters of the differential evolution method. In: Matoušek, R., Ošmera, P. (eds.) Proc. of Mendel 2002, 8th International Conference on Soft Computing, pp. 11–18 (2002)

[13] Liu, J., Lampinen, J.: Adaptive Parameter Control of Differential Evolution. In: Matoušek, R., Ošmera, P. (eds.) Proc. of Mendel 2002, 8th International Conference on Soft Computing, pp. 19–26 (2002)

[14] Liu, J., Lampinen, J.: A fuzzy adaptive differential evolution algorithm Soft Computing – A Fusion of Foundations. Methodologies and Applications 9(6), 448–462 (2005)

[15] Rönkkönen, J., Lampinen, J.: On using normally distributed mutation step length for the differential evolution algorithm. In: 9th Int. Conf. Soft Computing (MENDEL 2002), Brno, Czech Republic, June 5-7, 2002, pp. 11–18 (2003)

[16] Qin, A.K., Suganthan, P.N.: Self-adaptive differential evolution algorithm for numerical optimization. In: 2005 IEEE Congress Evolutionary Computation, Edinburgh, UK, September 2-5, vol. 2, pp. 1785–1791 (2005)

[17] Brest, J., Greiner, S., Bošković, B., Mernik, M., Žumer, V.: Self-Adapting Control Parameters in Differential Evolution, A Comparative Study on Numerical Benchmark Problems. IEEE Trans. on Evol. Comp. 10(6), 646–657 (2006)
[18] Schwefel, H.-P.: Numerical optimization of computer models. Wiley, New York (1981)
[19] Price, K.V., Rönkkönen, J.I.: Comparing the Uni-Modal Scaling Performance of Global and Local Selection in a Mutation-Only Differential Evolution Algorithm. In: IEEE Congress on Evolutionary Computation, 2006. CEC 2006, July 16-21, pp. 2034–2041 (2006)
[20] Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal 7, 308–313 (1965)
[21] Lampinen, J., Zelinka, I.: On Stagnation of the Differential Evolution Algorithm. In: Ošmera, P. (ed.) Proceedings of MENDEL 2000, 6th International Mendel Conference on Soft Computing, Brno, Czech Republic. Brno University of Technology, Faculty of Mechanical Engineering, June 7–9, pp. 76–83. Institute of Automation and Computer Science, Brno (Czech Republic) (2000)
[22] Goldberg, D.E.: Genetic algorithms in search optimization and machine learning. Addison-Wesley, Reading (1989)
[23] Karaboga, D., Ökdem, S.: A simple and global optimization algorithm for engineering problems: differential evolution algorithm. Turkish Journal of Electrical Engineering & Computer Sciences 12(1), 53–60 (2004)
[24] Das, S., Konar, A., Chakraborty, U.K.: Two improved differential evolution schemes for faster global search. In: Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO 2005), pp. 991–998 (2005)
[25] Das, S., Konar, A., Chakraborty, U.K.: Improved differential evolution algorithms for handling noisy optimization problems. In: Proc. IEEE Congress on Evolutionary Computation, Edinburgh (September 2005)
[26] Storn, R.: Digital Filter Design Program FIWIZ (2000), `http://www.icsi.berkeley.edu/~storn/fiwiz.html`
[27] Storn, R., Lampinen, J.: New DE Strategy, private Email communication (2000)
[28] Corana, A., Marchesi, M., Martini, C., Ridella, S.: Minimizing multimodal functions for continuous variables with the simulated annealing algorithm. ACM Transactions on Mathematical Software, 272–280 (March 1987)
[29] Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.: Opposition-Based Differential Evolution Algorithms. In: 2006 IEEE Congress on Evolutionary Computation, Vancouver, July 16-21, pp. 2010–2017 (2006)
[30] Ali, M.M.: Synthesis of the β-distribution as an aid to stochastic global optimization. Computational Statistics and Data Analysis (accepted for publication, 2006)
[31] Feoktistov, V.: Differential Evolution - In Search of Solutions. Springer, Heidelberg (2006)
[32] Onwubolu, G.C., Babu, B.V.: New Optimization Techniques in Engineering. In: Lampinen, J., Storn, R. (eds.) Differential Evolution, ch. 6, pp. 123–166. Springer, Heidelberg (2004)
[33] Martinek, P., Maršík, J.: Optimized Design of Analogue Circuits Using DE Algorithms. In: EDS 2005 IMAPS CS International Conference Proceedings, pp. 385–389. Vysoké učení technické v Brně, Brno (2005)
[34] Martinek, P., Tichá, D.: Analog Filter Design Based on Evolutionary Algorithms. In: AEE 2005 - Proceedings of the 4th WSEAS International Conference on: Applications of Electrical Engineering, vol. 1, pp. 111–115. WSEAS, Athens (2005)
[35] Vancorenland, P.J., De Ranter, C., Steyaert, M., Gielen, G.G.E.: Optimal RF design using smart evolutionary algorithms. In: Proceedings of 37th Design Automation Conference, Los Angeles, June 5-9, pp. 7–10 (2000)

[36] Francken, K., Vancorenland, P., Gielen, G.: DAISY: a simulation-based high-level synthesis tool for Delta Sigma modulators. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design. ICCAD 2000, San Jose, CA, USA, November 5-9, pp. 188–192 (2000)

[37] Storn, R.M.: System design by constraint adaptation and differential evolution. IEEE Transactions on Evolutionary Computation 3(1), 22–34 (1999)

[38] Storn, R.: On the usage of differential evolution for function optimization. In: Smith, M.H., Lee, M.A., Keller, J., Yen, J. (eds.) Proceedings of the North American Fuzzy Information Processing Society, pp. 519–523. IEEE Press, New York (1996)

[39] Report NDT3-04-2006: Differential Evolution for a Better Approximation to the Arctangent Function (April 26, 2006), http://www.nanodottek.com/Documents.htm

[40] Ursem, R.K., Vadstrup, P.: Parameter identification of induction motors using differential evolution. In: 2003 Congress on Evolutionary Computation, CEC 2003, vol. 2, pp. 790–796 (2003)

[41] Madisetti, V.K., Williams, D.B.: The Digital Signal Processing Handbook. Section VI, Adaptive Filtering. CRC Press, IEEE Press (1998)

[42] Murthy, C.S.R., Manoj, B.S.: Ad Hoc Wireless Networks: Architectures and Protocols. Prentice Hall, Englewood Cliffs (2004)

[43] Price, K., Storn, R., Lampinen, J.: Differential Evolution – A Practical Approach to Global Optimization. In: Chakraborty, N. (ed.) Genetic Algorithms and Related Techniques for Optimizing Si–H Clusters: A Merit Analysis for Differential Evolution, ch. 7.1. Springer, Berlin (2005)

[44] Price, K., Storn, R., Lampinen, J.: Differential Evolution – A Practical Approach to Global Optimization. In: Hancox, E.P., Derksen, R.W. (eds.) Optimization of an Industrial Compressor Supply System, ch. 7.3. Springer, Berlin (2005)

[45] Kasemir, K.U., Betzler, K.: Detecting ellipses of limited eccentricity in images with high noise levels. Image and Vision Computing 21(2), 221–227 (2003)

[46] Laskari, E.C., Meletiou, G.C., Vrahatis, M.N.: The Discrete Logarithm Problem as an Optimization Task: A First Study. In: Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA 2004) (IASTED 2004), Innsbruck, Austria. ACTA Press (2004) ISBN: 0-88986-375-X, ISSN: 1027-2666

[47] Laskari, E.C., Meletiou, G.C., Vrahatis, M.N.: Utilizing Evolutionary Computation Methods for the Design of S-boxes. In: Wang, Y., Cheung, Y.-m., Liu, H. (eds.) CIS 2006. LNCS (LNAI), vol. 4456. Springer, Heidelberg (2007)

[48] Henkel, W., Kessler, T.: Maximizing the Channel Capacity of Multicarrier Transmission by Suitable Adaptation of the Time-Domain Equalizer. IEEE Transactions on Communications 48(12), 2000–2004 (2000)

[49] Storn, R.: Differential Evolution – Ein praktischer Ansatz zur globalen Parameteroptimierung, Vortrag an der TU München, Seminar Elektronische Bauelemente (May 17, 2004)

[50] Onwubolu, G.C., Babu, B.V.: New Optimization Techniques in Engineering. Springer, Heidelberg (2004)

[51] Ruettgers, M.: Differential evolution: a method for optimization of real scheduling problems. Technical report at the International Computer Science Institute, TR-97-013, pp. 1–8 (1997)

[52] Babu, B.V., Rakesh, A.: A Differential Evolution Approach for Global Optimization of MINLP Problems. In: Proceedings of 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL 2002), Singapore, November 18-22, Paper No. 1033, vol. 2, pp. 880–884 (2002)

[53] Syslo, M.M., Deo, N., Kowalik, J.S.: Discrete optimization algorithms with Pascal programs. Prentice Hall, New Jersey (1983)
[54] Krink, T., Filipic, B., Fogel, G.B., Thomsen, R.: Noisy Optimization Problems - A Particular Challenge for Differential Evolution? In: Proceedings of 2004 Congress on Evolutionary Computation, pp. 332–339. IEEE Press, Piscataway (2004)
[55] Markon, S., Arnold, D.V., Baeck, T., Beielstein, T., Beyer, H.-G.: Thresholding - a selection operator for noisy ES. In: Proceedings of the 2001 Congress on Evolutionary Computation CEC 2001, May 27-30, pp. 465–472. IEEE Press, Piscataway (2001)
[56] Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.: Opposition-Based Differential Evolution for Optimization of Noisy Problems. In: 2006 IEEE Congress on Evolutionary Computation, IEEE World Congress on Computational Intelligence, Vancouver, July 16-21, pp. 1865–1872 (2006)
[57] Mendes, R., Mohais, A.S.: DynDE: a differential evolution for dynamic optimization problems. In: IEEE Congress on Evolutionary Computation, vol. 3, pp. 2808–2815 (September 2005)
[58] Crutchley, D.A., Zwolinski, M.: Using Evolutionary and Hybrid Algorithms for DC Operating Point Analysis of Nonlinear Circuits. In: Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002, Honolulu, Hawaii, May 12-17, vol. 1, pp. 753–758 (2002) ISBN 0-7803-7282-4
[59] Crutchley, D., Zwolinski, M.: Globally convergent algorithms for dc operating point analysis of nonlinear circuits. IEEE Transactions on Evolutionary Computing 7(1), 2–10 (2003)
[60] Crutchley, D.: Globally Convergent Algorithms for DC Operating Point Analysis of Nonlinear Electronic Circuits, PhD Dissertation, University of Southampton (2003)
[61] Antoniou, A.: Digital Filters – Analysis, Design, and Applications. McGraw-Hill, New York (1993)
[62] Dos Santos Coelhol, L., Mariani, V.C.: Combining of Differential Evolution and Implicit Filtering Algorithm Applied to Electromagnetic Design Optimization, Pontifical Catholic University of Parana, Technical Report
[63] Rogalsky, T., Derksen, R.W.: Hybridization of Differential Evolution for Aerodynamic Design. In: Proceedings of the 8th Annual Conference of the Computational Fluid Dynamics Society of Canada, June 11–13, pp. 729–736 (2000)
[64] Mydur, R.: Application of Evolutionary Algorithms & Neural Networks to Electromagnetic Inverse Problems. M.Sc. thesis, Texas A&M University, Texas, USA (2000)
[65] Nasimul Noman, N., Iba, H.: Enhancing Differential Evolution Performance with Local Search for High Dimensional Function Optimization. In: Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO 2005), pp. 967–974 (2005)
[66] Yuret, D., de la Maza, M.: Dynamic hill climbing: Overcoming the limitations of optimization techniques. In: The Second Turkish Symposium on Artifcial Intelligence and Neural Networks, pp. 208–212 (1993)
[67] Scales, L.E.: Introduction to non-linear optimization. Macmillan, London (1985)
[68] Chang, C.S., Xu, D.Y., Quek, H.B.: Pareto-optimal set based multi-objective tuning of fuzzy automatic train operation for mass transit system. IEEE Proceedings on Electric Power Applications 146(5), 577–583 (1999)
[69] Wang, F.-S., Sheu, J.-W.: Multi-objective parameter estimation problems of fermentation processes using a high ethanol tolerance yeast. Chemical Engineering Science 55(18), 3685–3695 (2000)

[70] Abbass, H.A., Sarker, R., Newton, C.: PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems. In: Proceedings of the 2001 congress on evolutionary computation, vol. 2, pp. 971–978. IEEE Press, Piscataway (2001)
[71] Abbass, H.A.: The self-adaptive Pareto differential evolution algorithm. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), Honolulu, Hawaii, May 2002, pp. 831–836 (2002b)
[72] Madavan, N.K.: Multiobjective optimization using a Pareto Differential Evolution approach. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), Honolulu, Hawaii, May 2002, pp. 1145–1150 (2002)
[73] Kukkonen, S., Lampinen, J.: GDE3: The third evolution step of generalized differential evolution. In: CEC 2005, Edinburgh, Scotland, pp. 443–450. IEEE Service Center (2005)
[74] Zielinski, K., Weitkemper, P., Laur, R., Kammeyer, K.D.: Examination of Stopping Criteria for Differential Evolution based on a Power Allocation Problem. In: 10th International Conference on Optimization of Electrical and Electronic Equipment, Brasov, Romania, May 18-19 (2006)
[75] Zaharie, D., Petcu, D.: Adaptive Pareto Differential Evolution and its Parallelization. In: Proc. of 5th International Conference on Parallel Processing and Applied Mathematics, Czestochowa, Poland (September 2003)
[76] Zaharie, D., Petcu, D.: Parallel implementation of multi-population differential evolution. In: Sinaia, R., Grigoras, D., et al. (eds.) Proc. of 2nd Workshop on Concurrent Information Processing and Computing (CIPC 2003) (2003)
[77] Tasoulis, D.K., Pavlidis, N.G., Plagianakos, V.P., Vrahatis, M.N.: Parallel Differential Evolution. In: Proceedings of the 2004 congress on evolutionary computation (CEC 2004), Portland OR, June 19-23, pp. 2023–2029 (2004)
[78] Kwedlo, W., Bandurski, K.: A Parallel Differential Evolution Algorithm. In: International Symposium on Parallel Computing in Electrical Engineering, 2006. PAR ELEC 2006, pp. 319–324 (2006)
[79] Angira, R., Babu, B.V.: Performance of modified differential evolution for optimal design of complex and non-linear chemical processes. Journal of Experimental & Theoretical Artificial Intelligence 18(4), 501–512 (2006)
[80] Goldstein, H.: Cure For The Multicore Blues. IEEE Spectrum, 36–39 (January 2007)