ON STAGNATION OF THE DIFFERENTIAL EVOLUTION ALGORITHM

Jouni Lampinen
Lappeenranta University of Technology
Department of Information Technology
Laboratory of Information Processing
P.O.Box 20, FIN-53851 Lappeenranta, Finland
phone: +358-5-6243430, fax: +358-5-6243456
E-mail: jlampine@lut.fi

Ivan Zelinka
Technical University of Brno
Faculty of Technology (Zlín)
Department of Automatic Control
nám. T.G.M. 275, 762 72 Zlín, Czech Republic
phone: +420-67-721 15 21, fax: +420-67-721 15 21
E-mail: Zelinka@zlin.vutbr.cz

Abstract: *This article discusses the stagnation of an evolutionary optimization algorithm called Differential Evolution. Stagnation problem refers to a situation in which the optimum seeking process stagnates before finding a globally optimal solution. Typically, stagnation occurs virtually without any obvious reason. The stagnation differs from the premature convergence so that the population remains diverse and unconverged after stagnation, but the optimization process does not progress anymore. The reasons for this problem have remained unknown so far. This article uncovers this problem describing the basic nature of stagnation phenomena, a mechanism behind it and some reasons for stagnation. Advices for reducing the risk of stagnation are concluded on basis of the new findings.*

Keywords: evolutionary algorithms, differential evolution, non-linear optimization, premature convergence, stagnation

1 Introduction

In general, it is commonly known that non-linear global optimization algorithms often converge prematurely to a point that is not globally optimal. Typically, when a premature convergence of an evolutionary optimization algorithm occurs:

- The population has converged to local optima of a multimodal objective function.
- The population has lost its diversity.
- The search algorithm proceeds slowly or does not proceed at all.

In case of **Differential Evolution** (DE) algorithm [SP95a, SP97a, SP97b, Pri99], our practical experiences have suggested that the DE may occasionally stop proceeding towards the global optimum virtually without any obvious reason, while:

- The population has not converged to a local optimum or any other point.
- The population is still remaining diverse.
- Occasionally, even new individuals may enter to the population, but the algorithm does not progress by finding any better solutions.

This situation is called **stagnation** in order to distinguish between it and **premature convergence**. The mentioned properties can be used to detect which one has occurred. In both cases, the algorithm has lost its capability of finding better solutions for the current problem.

Since the stagnation of DE-algorithm has remained as an unexplained phenomena, it is important to understand what happens and why, when the DE stagnates. So far, it is known that if a small population size (<<20) is applied, the stagnation is more likely to occur, while larger populations are not prone to stagnate.

It should be noticed here, that stagnation does not seem to be a severe problem in practice, since a high number of applications of DE-algorithm have been reported [Lam99c], and none of them has complained about stagnation problems. More typically, findings suggesting effectiveness, high convergence rate and robustness of DE were reported. However, solving the stagnation problem may still allow working with smaller populations in future and thus result in a higher convergence velocity or alternatively, improve the robustness.

In the following, after explaining the principle of the DE-algorithm, it is demonstrated that the stagnation is a relatively simple phenomena that may happen due to the fact that the population of DE-algorithm have potential for generating only a limited number of different trial solutions within one generation. In future, recognition of this fact will hopefully lead to a redesigned DE-algorithm which is capable of overcoming the stagnation problem.

2 Differential Evolution Algorithm

Price and Storn first introduced the DE-algorithm a few years ago [SP95a, SP97, SP97b, Pri99]. DE can be classified as an *evolutionary optimization algorithm.* The algorithm was originally designed to operate on continuous variables. Recently we have extended DE also for handling *mixed integer-discrete-continuous* variables [LZ99a, LZ99b, LZ99c, LZ99d]. At present, there are several variants of DE [Pri99]. The particular version used throughout this investigation is the *DE/rand/1/bin* scheme. However, the major findings reported here can be applied to other schemes, too, while this particular scheme is used here as an example.

Generally, the function to be optimized, *f*, is of the form:

$$f(X): R^D \rightarrow R \tag{1}$$

The optimization goal is to minimize the value of this *objective function f(X)* by optimizing the values of its parameters:

$$X = (x_1, \ldots, x_D), \quad X \in R^D \tag{2}$$

where *X* denotes a vector composed of *D* objective function parameters. Usually, the parameters of the objective function are also subject to lower and upper boundary constraints, $x^{(L)}$ and $x^{(U)}$, respectively:

$$x_j^{(L)} \le x_j \le x_j^{(U)} \qquad j = 1, \ldots, D \tag{3}$$

As with all evolutionary optimization algorithms, DE operates on a *population*, P_G, of candidate solutions, not just a single solution. These candidate solutions are the *individuals* of the population. In particular, DE maintains a population of constant size that consists of *NP*, real-valued vectors, X_{iG}, where *i* indexes the population and *G* is the *generation* to which the population belongs.

$$P_G = X_{i,G} \qquad i = 1, \ldots, NP, \quad G = 1, \ldots, G_{max} \tag{4}$$

Additionally, each vector contains *D* real parameters (*chromosomes* of individuals):

$$X_{i,G} = x_{j,i,G} \qquad i = 1, \ldots, NP, \quad j = 1, \ldots, D \tag{5}$$

In order to establish a starting point for optimum seeking, the population must be initialized. Often there is no more knowledge available about the location of a global optimum than the limits of the problem variables. In this case, a natural way to seed the initial population, $P_{G=0}$, is with random values chosen from within the given boundary constraints:

$$P_0 = x_{j,i,0} = rand_j[0,1] \cdot \left(x_j^{(U)} - x_j^{(L)}\right) + x_j^{(L)} \qquad i = 1, \ldots, NP, \quad j = 1, \ldots, D \tag{6}$$

where $rand_j[0,1]$ denotes a uniformly distributed random value within the range: [0.0,1.0] that is chosen anew for each *j*.

DE's self-referential population reproduction scheme is different from other evolutionary algorithms (see Figure 1). From the 1st generation forward, vectors in the current population, P_G, are randomly sampled and combined to create candidate vectors for the subsequent generation, P_{G+1}. The population of candidate, or "trial" vectors, $P'_{G+1} = U_{i,G+1} = u_{j,i,G+1}$, is generated as follows:

$$u_{j,i,G+1} = \begin{cases} v_{j,i,G+1} = x_{j,r3,G} + F \cdot \left(x_{j,r1,G} - x_{j,r2,G}\right) & \text{if} \quad rand_j[0,1) \le CR \vee j = k \\ x_{j,i,G} & \text{otherwise} \end{cases} \tag{7}$$

where

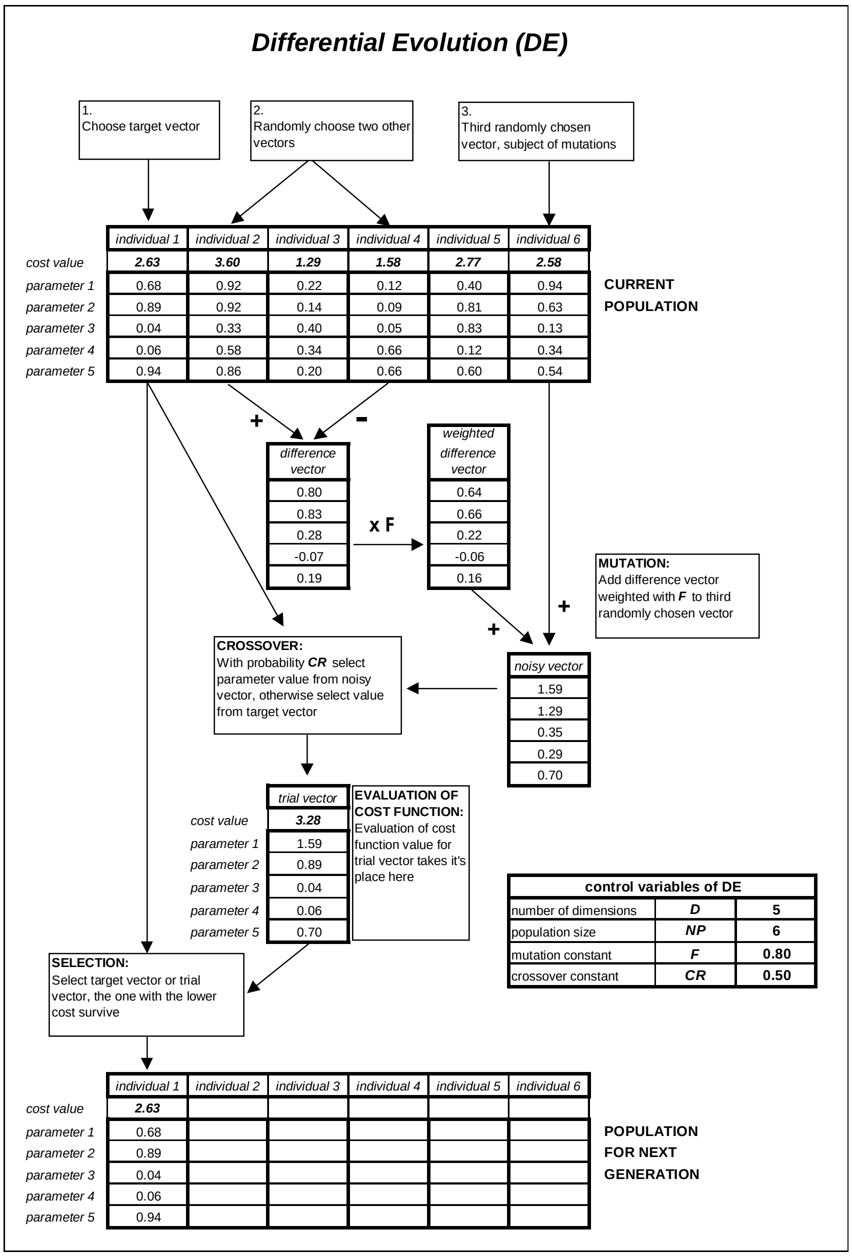
$i = 1, \ldots, NP, \quad j = 1, \ldots, D$

$k \in \{1, \ldots, D\}$, random parameter index, chosen once for each i

$r_1, r_2, r_3 \in \{1, \ldots, NP\}$, randomly selected, except: $r_1 \ne r_2 \ne r_3 \ne i$

$CR \in [0,1], \quad F \in (0,1+]$

The randomly chosen indexes, r_1, r_2 and r_3 are different from each other and also different from the running index, i. New, random, integer values for r_1, r_2 and r_3 are chosen for each value of the index i, i.e., for each individual. The index, k, refers to a randomly chosen chromosome which is used to ensure that each individual trial vector, $U_{i,G+1}$, differs from its counterpart in the previous generation, X_{iG}, by at least one parameter. A new, random, integer value is assigned to k prior to the construction of each trial vector, i.e., for each value of the index i.


Differential Evolution (DE)
1. Choose target vector
2. Randomly choose two other vectors
3. Third randomly chosen vector, subject of mutations
CURRENT POPULATION
+
-
x F
+
+
POPULATION FOR NEXT GENERATION

	individual 1	individual 2	individual 3	individual 4	individual 5	individual 6
cost value	**2.63**	**3.60**	**1.29**	**1.58**	**2.77**	**2.58**
parameter 1	0.68	0.92	0.22	0.12	0.40	0.94
parameter 2	0.89	0.92	0.14	0.09	0.81	0.63
parameter 3	0.04	0.33	0.40	0.05	0.83	0.13
parameter 4	0.06	0.58	0.34	0.66	0.12	0.34
parameter 5	0.94	0.86	0.20	0.66	0.60	0.54

difference vector	weighted difference vector
0.80	0.64
0.83	0.66
0.28	0.22
-0.07	-0.06
0.19	0.16

MUTATION: Add difference vector weighted with ***F*** to third randomly chosen vector

noisy vector
1.59
1.29
0.35
0.29
0.70

CROSSOVER: With probability ***CR*** select parameter value from noisy vector, otherwise select value from target vector

	trial vector
cost value	**3.28**
parameter 1	1.59
parameter 2	0.89
parameter 3	0.04
parameter 4	0.06
parameter 5	0.70

EVALUATION OF COST FUNCTION: Evaluation of cost function value for trial vector takes it's place here

control variables of DE		
number of dimensions	***D***	**5**
population size	***NP***	**6**
mutation constant	***F***	**0.80**
crossover constant	***CR***	**0.50**

SELECTION: Select target vector or trial vector, the one with the lower cost survive

	individual 1	individual 2	individual 3	individual 4	individual 5	individual 6
cost value	**2.63**					
parameter 1	0.68					
parameter 2	0.89					
parameter 3	0.04					
parameter 4	0.06					
parameter 5	0.94					

Figure 1. *Differential Evolution works directly with the floating-point valued variables of the objective function, not with their (binary) encoding. The functioning of DE is here illustrated in the case of a simple objective function* $f(X) = x_1 + x_2 + x_3 + x_4 + x_5$.

F and *CR* are DE control parameters. Like *NP*, both values remain constant during the search process. *F* is a real-valued factor in the range (0.0,1.0+] that scales the differential variations. The upper limit on *F* has been empirically determined.

CR is a real-valued crossover factor in range [0.0,1.0] that controls the probability that a trial vector parameter will come from the randomly chosen, mutated vector, $v_{j,i,G+1}$, instead of from the current vector, $x_{j,i,G}$. Generally, both *F* and *CR* affect the convergence velocity and robustness of the search process. Their optimal values are dependent both on objective function characteristics and on the population size, *NP*. Usually, suitable values for

F, *CR* and *NP* can be found by trial-and-error after a few tests using different values. Practical advice on how to select control parameters *NP*, *F* and *CR* can be found in [SP95a, Sto96a, SP97a, SP97b, Pri99], for example.

DE's selection scheme also differs from other evolutionary algorithms. The population for the next generation, P_{G+1}, is selected from the current population, P_G, and the child population, according to the following rule:

$$X_{i,G+1} = \begin{cases} U_{i,G+1} = u_{j,i,G+1} & \text{if} \quad f(U_{i,G+1}) \le f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases} \tag{8}$$

Thus, each individual of the temporary population is compared with its counterpart in the current population. Assuming that the objective function is to be minimized, the vector with the lower objective function value wins a place in the next generation's population. As a result, all the individuals of the next generation are as good or better than their counterparts in the current generation. The interesting point concerning DE's selection scheme is that a trial vector is only compared to one individual, not to all the individuals in the current population.

3 Stagnation – an example

This Section describes a mechanism on how the stagnation may happen by a simple example. It is shown that the underlying mechanism for stagnation may be relatively simple.

Consider the following example. The problem to be solved is two-dimensional, having two parameters x_1 and x_2. The DE control parameter settings are *NP*=4, *F*=0.5 and *CR*=1.0. The current population consisting of four individuals *A, B, C* and *D,* is given in Table 1 and illustrated in Figure 2. The next step is to create the population of the subsequent generation by applying Equations 7 and 8.

In the following, the number of possible trial solutions, that the reproduction operation of DE (Equation 7) is capable of generating on basis of the current population, is referred as a **number of potential trial solutions** n_{trial}. Table 2 enumerates all the 24 potential trial solutions that Equation 7 may create in case of the current example. These are the possible candidate solutions for the population of the following generation – simply, there are no other points to replace any member of the current population.

Table 1. *The current population of DE for the example.*

The Current Population				
Parameter	Individual vectors			
	A	B	C	D
x_1	3.92	-4.88	-2.43	-2.72
x_2	0.31	-4.23	-4.48	4.87

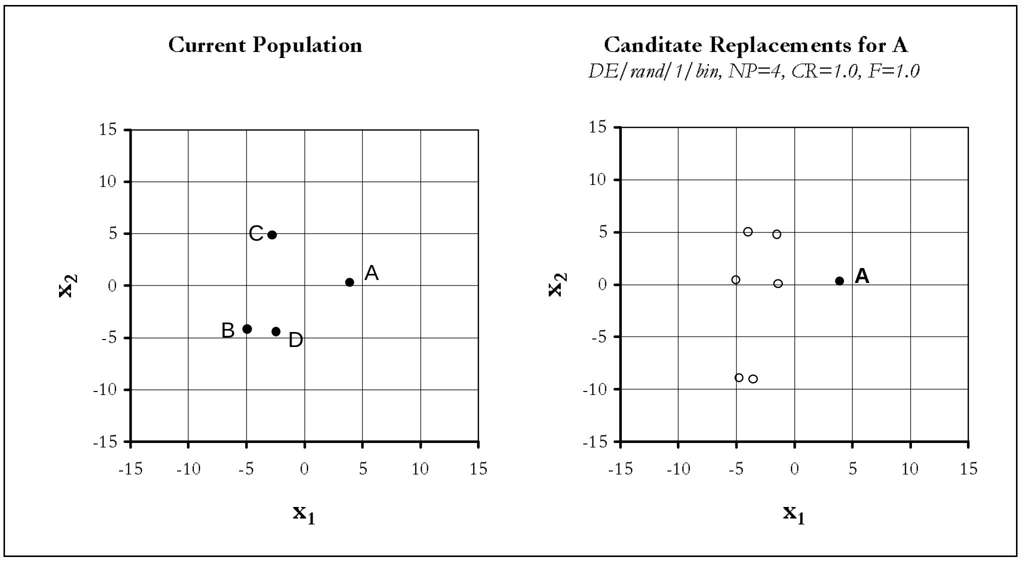


Figure 2. *The current population (left) and all possible points that may replace individual A and can be generated by applying the Equation 7 are shown (right). Note that by applying the Equation 7, only a finite number of potential trial points can be generated.*

Table 2. *All the potential trial solutions that the DE-algorithm may generate on basis of the current population (Table 1) with the settings NP*=4, *F*=0.5 *and CR*=1.0.

All possible points that the reproduction operation can generate *										
All possible vector differentials			All possible reproduction results							
			differential * *F* + vector *A*		differential * *F* + vector *B*		differential * *F* + vector *C*		differential * *F* + vector *D*	
vectors	diff. for x_1	diff. for x_2	x_1	x_2	x_1	x_2	x_1	x_2	x_1	x_2
A-A	**	**	**	**	**	**	**	**	**	**
A-B	4.40	2.27	**	**	**	**	1.97	-2.21	1.68	7.14
A-C	3.18	2.39	**	**	-1.70	-1.83	**	**	0.46	7.27
A-D	3.32	-2.28	**	**	-1.56	-6.51	0.89	-6.76	**	**
B-A	-4.40	-2.27	**	**	**	**	-6.83	-6.75	-7.12	2.60
B-B	**	**	**	**	**	**	**	**	**	**
B-C	-1.22	0.13	2.70	0.44	**	**	**	**	-3.94	5.00
B-D	-1.08	-4.55	2.84	-4.24	**	**	-3.51	-9.03	**	**
C-A	-3.18	-2.39	**	**	-8.06	-6.62	**	**	-5.90	2.48
C-B	1.22	-0.13	5.15	0.18	**	**	**	**	-1.49	4.75
C-C	**	**	**	**	**	**	**	**	**	**
C-D	0.14	-4.68	4.07	-4.37	-4.74	-8.90	**	**	**	**
D-A	-3.32	2.28	**	**	-8.20	-1.95	-5.75	-2.20	**	**
D-B	1.08	4.55	5.00	4.86	**	**	-1.35	0.07	**	**
D-C	-0.14	4.68	3.78	4.99	-5.02	0.45	**	**	**	**
D-D	**	**	**	**	**	**	**	**	**	**

* Assumed: Current population as in Table 1, *DE/rand/1/bin* scheme, *NP*=4, *CR*=1.0, *F*=0.5

** Impossible vector combination according to Equation 7. All vectors used for generating a trial solution should be mutually different from each other and also different from the running index (current individual). See Equation 7 and Figure 1.

Only 6 out of these 24 potential trial solutions may replace the vector *A* (Figure 2). Consider the situation that all of these potential points result in a worse objective function value than the vector *A*. Thus, due to the selection rule (Equation 8) the original point *A* will certainly be copied to the next generation in any case within possibilities, since it returns a better objective function value than any of the trial points that are potential to replace *A* (see Figure 2).

What if this situation will be repeated also with the vectors *B*, *C* and *D*? It means that the next generation would be identical to the current one, so also it will not be able to generate a solution, which would be capable of replacing any member of the current population. So, this situation is permanent – the **search will stagnate**, while the **population is not converged** at all. Also the population is **still diverse**, but not capable of any more progress.

4 The role of DE-control variables

On basis of the example, it is obvious that the risk of stagnation is (at least approximately) inversely proportional to the number of potential trial solutions n_{trial} that the algorithm is capable of generating within one generation. The higher n_{trial}, the lower risk to stagnate. The number of potential trial solutions in case of *DE/rand/1/bin* depends on the DE control variables as follows:

$$n_{trial} = \begin{cases} NP^3 - 3 \cdot NP^2 + 2 \cdot NP & \text{if} \quad CR = 1 \\ \left(NP^3 - 3 \cdot NP^2 + 2 \cdot NP\right) \cdot D \cdot NP & \text{if} \quad CR = 0 \\ \left(NP^3 - 3 \cdot NP^2 + 2 \cdot NP\right) \cdot 2^D \cdot NP & \text{otherwise} \end{cases} \tag{9}$$

Note here, that the n_{trial} is the maximum number of **different** trial solutions, and the actual number of **different** potential trial solutions depends on *F* and the current population. For example setting *F*=1.0 result in coincidental potential trial points (duplicate instances) as demonstrated in Figure 3, since *F(B-C)+D=F(D-C)+B* if *F*=1.0.

Thus, the factors determining the risk of stagnation are the control parameters of *NP*, *CR*, *F*, the current population and the objective function. Assuming that the objective function is given and the current population depends on a series of random events, only *NP*, *CR* and *F* are under control of the user. In the following, each of them will be discussed more closely.

4.1 *Mutation amplification – F*

According Equation 9, *NP* and *CR* determine the number of potential trial solutions. How many of them are different from each other, depends on the current population and *F*. Figure 3 compares all possible candidate solutions with using *F*=0.5 (as in Table 2) and using *F*=1.0. Setting *F*=1.0 is a special case where the number of **different** potential candidate solutions is reduced since each potential trial point appears twice. So it cannot be recommended to set *F* **exactly** to 1.0 from stagnation point of view, for example, *F*=0.99 could be used instead.

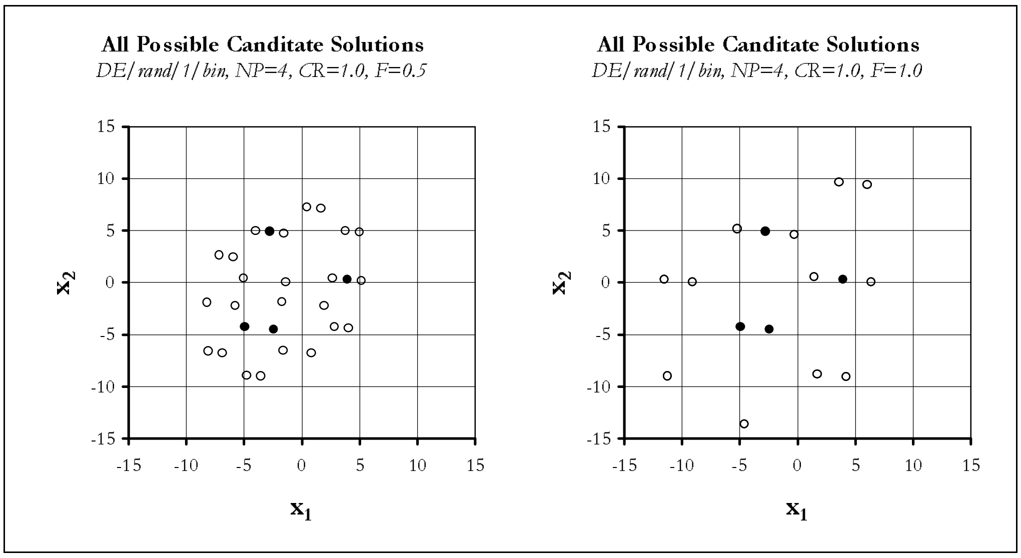


Figure 3. *Comparing different values for differential variation factor F. All potential trial solutions are shown. F*=1.0 *results in a duplicate instances of potential trial solutions. In this case F*=0.5 *will provide* 24 *different potential trial solutions while F*=1.0 *also provide* 24 *potential trial solutions, but only* 12 ***different*** *solutions.*

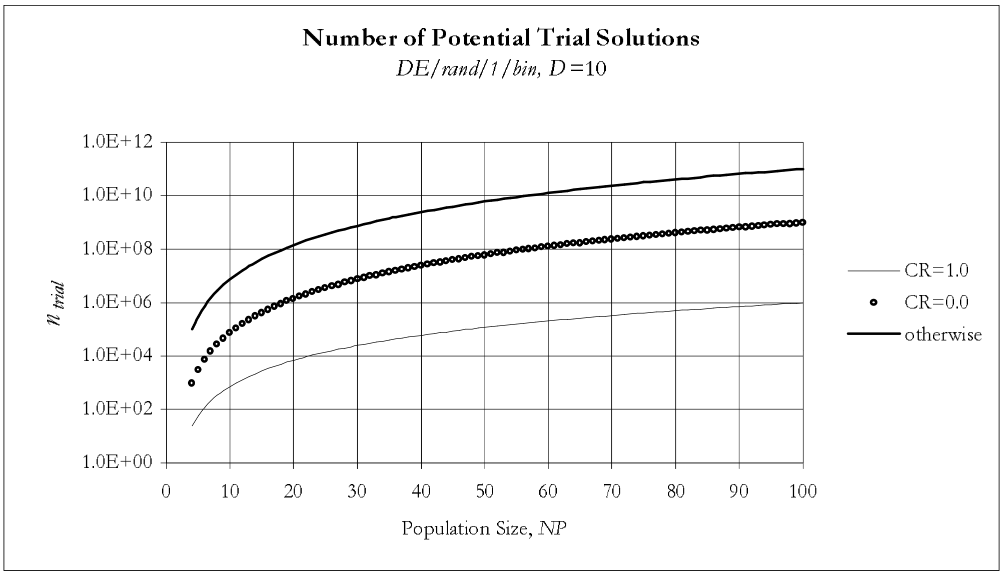


Figure 4. *The number of potential trial solutions as a function of population size in case of a* 10*-parameter objective function. A relatively low population size may provide a finite, but still huge number of potential trial solutions. However, setting CR*=1.0 *results in a significantly reduced number of potential trial solutions.*

4.2 *Crossover factor – CR*

In the case of *CR*=1.0 the DE-algorithm is rotationally invariant [Pri99], so this setting is generally justified as such [Sal96]. More specifically, algorithms that accommodate rotational invariance are the only justified ones, unless one has *a priori* knowledge stating that the objective function is separable [Sal96, Pri99]. However, in case of *CR*=1.0, the crossover does not increase the number of potential solutions since all the components for the trial vector come from the noisy vector (see Figure 1 and Equation 9). Using a slightly lower CR, for example 0.99, will increase the number of potential solutions dramatically, since then also combinations of the target vector and the noisy vector are within possibilities. This should reduce the risk of permanent stagnation by increasing the number of potential trial solutions. So, setting *CR* **exactly** to 1.0 cannot be recommended.

4.3 Population size – NP

A larger population size reduces not only the risk of premature convergence, but the risk of stagnation as well. The number of potential trial solutions will increase quickly with the population size (Equation 9), which explains why the stagnation typically occurs only with a small population size. However, a larger population size results in a lower convergence velocity.

5 Conclusions

A reason for stagnation of DE algorithm is that the reproduction operation (Equation 7) is capable of providing only a finite number of potential trial solutions (Equation 9, Figure 4). If none of them is capable of replacing a member of the current population during the selection operation (Equation 8) the algorithm will stagnate. The probability of stagnation depends on how many different potential trial solutions are available and also their capability to enter into the population of the following generation. Thus, it depends on:

- Population size, *NP*
- Mutation amplification, *F*
- Crossover factor, *CR*
- Current population
- Objective function

A larger population reduces the risk of stagnation – the stagnation is unlikely to occur with a larger population size (>20). Usage of *F*=1.0 cannot be recommended, since it will result in multiple instances of identical potential trial vectors reducing the number of different potential trial solutions. The number of potential trial solutions will be reduced dramatically if *CR*=1.0 is used. From stagnation point of view, setting either *CR* or *F* exactly to value 1.0 cannot be recommended, however, for example the value 0.99 can be used instead.

Currently, numerous investigations have suggested effectiveness, efficiency and robustness of DE-algorithm [Lam99c]. However, the findings discussed in this article are suggesting some further work, since solving the stagnation problem would undoubtedly result in an improved DE-algorithm, with even better performance.

Acknowledgements

The authors would like to thank Kenneth V. Price for the time he devoted for going through the first draft version of this article. Especially we would like to thank Ken for sharing his knowledge, including his constructive comments, which helped us to straighten and improve the final manuscript. Also we would like to thank Mrs. Junhong Liu for her kind help with the proofreading of this article.

References

[Lam99c] Lampinen, Jouni (1999). *A Bibliography of Differential Evolution Algorithm*. Technical Report. Lappeenranta University of Technology, Department of Information Technology, Laboratory of Information Processing, 16th October 1999. Available via Internet `http://www.lut.fi/~jlampine/debiblio.htm`.

[LZ99a] Jouni Lampinen – Ivan Zelinka (1999). *Mechanical Engineering Design Optimization by Differential Evolution*. In: David Corne, Marco Dorigo and Fred Glover (editors) (1999). New Ideas in Optimization. McGraw-Hill, London (UK), pp. 127–146. ISBN 007-709506-5.

[LZ99b] Jouni Lampinen – Ivan Zelinka (1999). *Mixed Integer-Discrete-Continuous Optimization By Differential Evolution, Part 1: the optimization method.* In: Ošmera, Pavel (ed.) (1999). Proceedings of MENDEL'99, 5th International Mendel Conference on Soft Computing, June 9.–12. 1999, Brno, Czech Republic. Brno University of Technology, Faculty of Mechanical Engineering, Institute of Automation and Computer Science, Brno (Czech Republic), pp. 71–76. ISBN 80-214-1131-7.

[LZ99c] Jouni Lampinen – Ivan Zelinka (1999). *Mixed Integer-Discrete-Continuous Optimization By Differential Evolution, Part 2: a practical example*. In: Ošmera, Pavel (ed.) (1999). Proceedings of MENDEL'99, 5th International Mendel Conference on Soft Computing, June 9.–12. 1999, Brno, Czech Republic. Brno University of Technology, Faculty of Mechanical Engineering, Institute of Automation and Computer Science, Brno (Czech Republic), pp. 77–81. ISBN 80-214-1131-7.

[LZ99d] Jouni Lampinen – Ivan Zelinka (1999). *Mixed Variable Non-Linear Optimization by Differential Evolution*. In: Zelinka, Ivan (ed.) (1999). Proceedings of Nostradamus'99, 2nd International Prediction Conference, October 7.–8. 1999, Zlin, Czech Republic. Technical University of Brno, Faculty of Technology Zlin, Department of Automatic Control, Zlin (Czech Republic), pp. 45–55. ISBN 80-214-1424-3.

[Pri99] Price, Kenneth V. (1999). *An Introduction to Differential Evolution*. In: David Corne, Marco Dorigo and Fred Glover (editors) (1999). New Ideas in Optimization. McGraw-Hill, London (UK), pp. 79–108. ISBN 007-709506-5.

[Sal96] Salomon, Ralf (1996). *Reevaluating Genetic Algorithm Performance under Coordinate Rotation of Benchmark Functions*. BioSystems, 39(3):263–278, Elsevier Science.

[Sto96a] Storn, Rainer (1996). *On the usage of differential evolution for function optimization*. In: 1996 Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS 1996), Berkeley, pp. 519–523. PostScript-file available from `http://www.icsi.berkeley.edu/~storn/litera.html`.

[SP95a] Storn, Rainer and Price, Kenneth (1995). *Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces*. Technical Report TR-95-012, ICSI, March 1995. Available via `ftp` from `ftp.icsi.berkeley.edu/pub/techreports/1995/tr-95-012.ps.Z`.

[SP97a] Storn, Rainer and Price, Kenneth (1997). *Differential Evolution – A simple evolution strategy for fast optimization*. Dr. Dobb's Journal, April 97, pp. 18–24 and p. 78.

[SP97b] Storn, R. and Price, K. (1997). *Differential Evolution – a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces*. Journal of Global Optimization, 11(4):341–359, December 1997. Kluwer Academic Publishers.

Reference data for this document:

[LZ2000] Jouni Lampinen and Ivan Zelinka (2000). *On Stagnation of the Differential Evolution Algorithm*. In: Ošmera, Pavel (ed.) (2000). Proceedings of *MENDEL 2000, 6th International Mendel Conference on Soft Computing, June 7.–9. 2000, Brno, Czech Republic*. Brno University of Technology, Faculty of Mechanical Engineering, Institute of Automation and Computer Science, Brno (Czech Republic), pp. 76–83. ISBN 80-214-1609-2.