Towards Self–Adapting Evolution Strategies

Frank Kursawe
Systems Analysis Research Group
Department of Computer Science
University of Dortmund,
D-44221 Dortmund, Germany
e-mail: kursawe@LS11.informatik.uni-dortmund.de

ABSTRACT

Optimization algorithms imitating certain principles of nature have proved their capability in various domains of applications. Dealing with parameter optimization problems one usually trades the original problem for a much simpler one, estimating the exogenous parameters of the algorithm chosen to yield a good solution as fast as possible. On the one hand, this paper demonstrates empirically for a small set of test functions, how convergence velocity and reliability of evolution strategies depend on the recombination operator chosen. On the other hand, first results indicate that the capability of self–adaptation within evolution strategies may be exploited in order to reduce the number of exogenous parameters, thus leading to more robust strategies.

1. Introduction

Based upon a Darwin's observations and the modern synthetic theory of organic evolution, *evolution strategies* (ESs) represent a powerful tool for parameter optimization tasks, because they do not rely on mathematical prerequisites like e. g. analysis does. The more difficult a problem is the more appropriate (and sometimes necessary) the use of evolutionary algorithms becomes. A *population* of *individuals* each of which representing one point of the solution space collectively evolves towards better solutions by means of the randomized operators *recombination* and *mutation* and the deterministic *selection*. Recombination allows the exchange of already existing genes, mutation introduces new genetic material and the selection step guarantees that individuals with a lower fitness value, measured in terms of the objective function $f : D \subseteq \boldsymbol{R}^n \longrightarrow \boldsymbol{R}$, will no longer reproduce in the next generation. For details of the algorithm, see e. g. [1]. Although originally developed for experimental problems [4], nowadays, the multi–membered ES, as introduced by Schwefel [6], is most often used to solve continuous parameter optimization problems. Learning or adaptation takes place *collectively* and on two levels: Object variables x_i and strategy variables σ_i determining the mutability of the x_i have to adapt together, because the selection operator implicitly evaluates the strategy variables, too.

The other well–known representative of the class of *evolutionary algorithms* should at least briefly be mentioned here: Individuals within *genetic algorithms* represent bitstrings of a fixed length, thus being more appropriate for discrete optimization problems. For a good survey, see e. g. [2].

The results of the test runs which will be presented in the following have been achieved using three functions:

$$\begin{array}{lll} \text{sphere model (1.1)} & : & \sum_{i=1}^{n} x_i^2 \\ \text{Schwefel's problem 1.2 [6]} & : & \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \\ \text{scaling test} & : & \sum_{i=1}^{n} \left(i \cdot x_i^2 \right) \end{array}$$

The following coding is used in order to distinguish the different types of recombination both numerically and symbolically. The numerical code is only necessary for the plotting program, the corresponding symbolic one is more intuitive than digits. In every plot, the first digit represents the recombination type of the object variables, the second represents the type of recombination used for the strategy variables and the third digit has been introduced to characterize the recombination of covariances in the future. It is not relevant in this paper.

no recombination	1	–
discrete, 2 parents	2	d
discrete, μ parents	3	D
intermediate, 2 parents	4	i
intermediate, μ parents	5	I
1–point crossover	6	c(1)
2–point crossover	7	c(2)

Unless a figure exhibits single runs in order to demonstrate a typical effect, the average convergence velocity $\log_{10}(f_{start}/f_{best-occurred})$ of 100 runs is displayed.

2. Surprising(?) Observations

In the early days of evolution strategies, the algorithm resembled a person turning some knobs on the outside of a black box. No closed form analytical objective function was available, e. g. when trying to optimize the flow through a nozzle. If nowadays any optimization algorithm claiming to be *robust* wants to live up to the expectation created, it ought to be capable of dealing with this sort of situation, e. g. when optimizing a simulation model. Finding ‘reasonable’ starting values for the algorithm’s parameters should suffice as far as convergence reliability is concerned. Instead of a black box we started off with, we may regard the algorithm as a grey box with a reduced amount of ‘knobs’ or parameters. A robust algorithm should trade the original, high–dimensional problem for a much simpler one — maximizing convergence velocity.

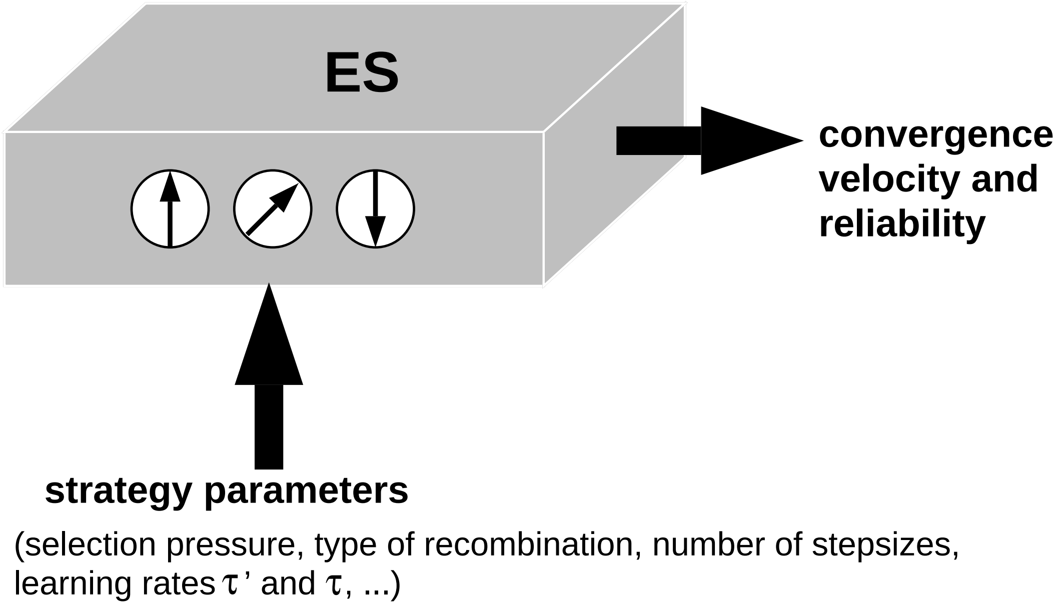
ES
convergence velocity and reliability
strategy parameters
(selection pressure, type of recombination, number of stepsizes, learning rates τ' and τ, ...)

Fig. 1: Solving one parameter optimization leads to another parameter optimization problem.

But how can we decide which values are reasonable if left in a black box–type of situation? The theory developed so far gives little guidance for the appropriate setting of the control ‘knobs’. Even more, do we know enough about those objective functions which are widely used when comparing the performance of various optimization algorithms? Do we really face a glass box–type of situation if we use the well–known sphere model as the objective function? Can we expect the same behaviour over a broad range of dimensions? And is convergence velocity really the only thing to worry about dealing with simple and unimodal functions?

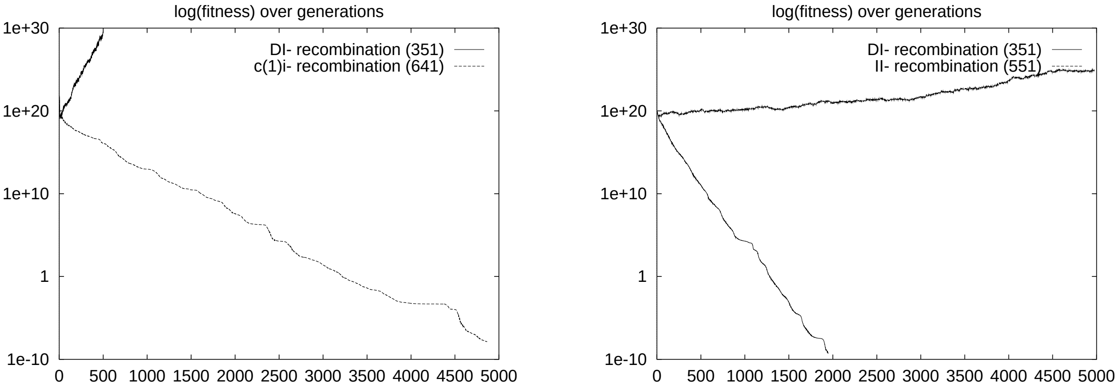

Fig. 2: (10, 100)–ESs, Schwefel's problem 1.2, $n = n_\sigma = 30$ and (15, 100)–ESs, sphere model, $n = n_\sigma = 100$

Maybe, the divergent run in Figure 2 (left) does not come as a big surprise to people being familiar with this function. Trying to stay on the ridge with a discrete recombination of the object variables is simply not appropriate. But what about the high–dimensional sphere model? As Figure 2 (right) demonstrates, a setting of the recombination operator also exists which leads to divergence. This setting is recommended as a useful default in [1].

Let us examine these effects more detailed now by comparing all types of recombination possible with respect to their effect on convergence velocity and reliability.

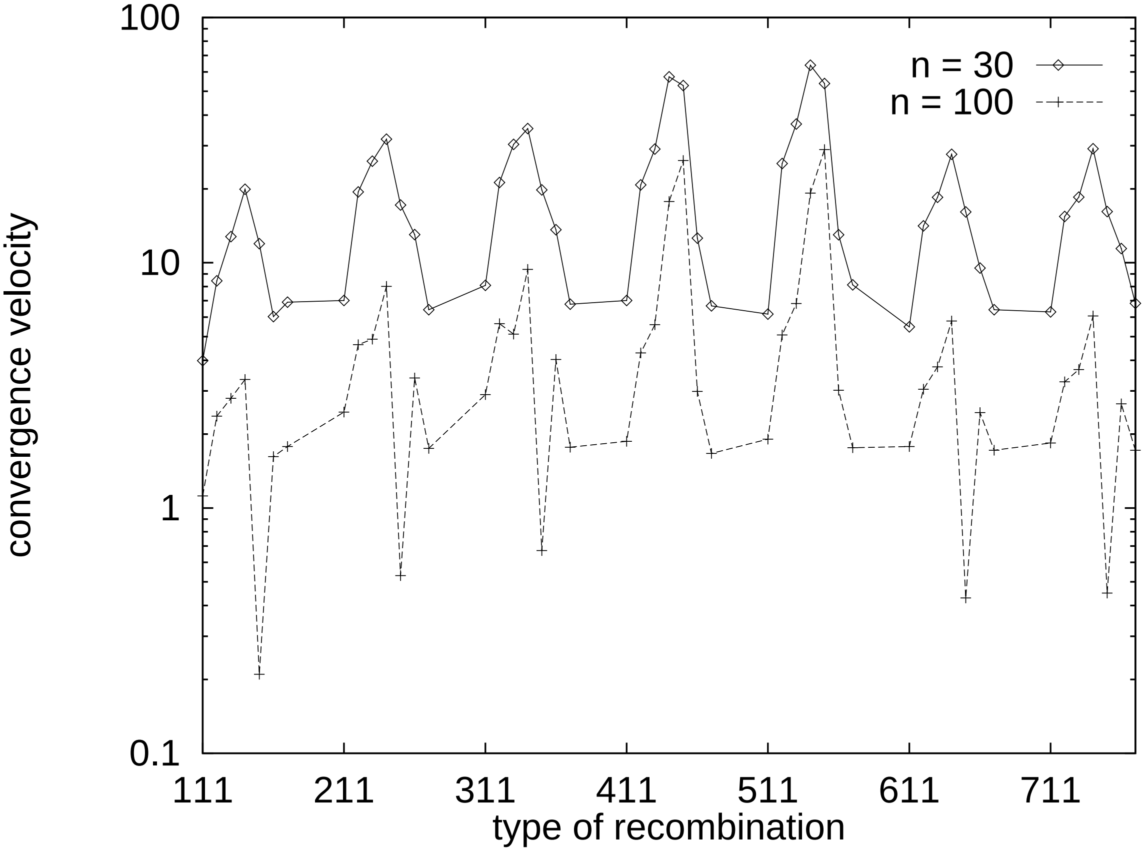

Fig. 3: (15, 100)–ES, sphere model, $n_\sigma = n$

The upper curve in Figure 3 represents a behaviour one would expect of a robust optimization algorithm. For $n = 30$ parameters, the ES only 'punishes' the non–optimal choice of the recombination operator by a slower convergence. But the negative peaks in the lower curve ($n = 100$) do no longer indicate slow convergence, but rather *divergence*, because the best value which has occurred during each run is compared to the starting value. After a short phase of progress in the beginning, the algorithm diverges. If we invest more knowledge about the sphere model, though, by exploiting its symmetry using only one stepsize (mathematically correct: standard deviation) for all object variables x_i, the ES returns to a robust behaviour again.

3. Self–Adaptation of n_σ

3.1. Algorithm

As a first step towards finding the best ES for a given problem, the number of stepsizes n_σ is allowed to vary during the run according to the following algorithm.

With probability $p = 0.05$ the number of stepsizes within an individual changes. Reducing or increasing this number by one is equally probable. If necessary, an additional stepsize is set to $\sigma_{n_\sigma} := 1/\left(n_\sigma - 1\right) \cdot \sum_{i=1}^{n_\sigma - 1} \sigma_i$.

3.2. Results

How does the modified algorithm perform against the best variants of the standard $(\mu\,,\,\lambda)$–ES as far as the recombination operator is concerned? Let us regard the sphere model (see Figure 4 left) and the scaling test function (see Figure 4 right), where one stepsize or n stepsizes are known to yield the fastest convergence, respectively.

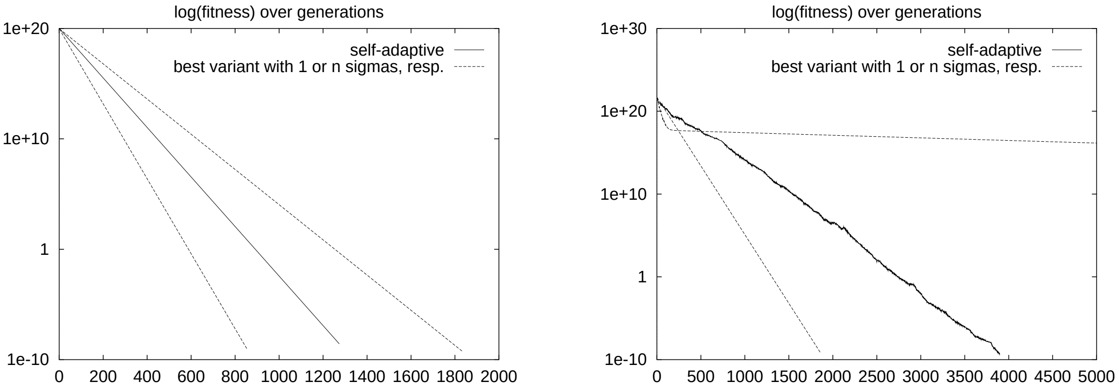

Fig. 4: (15, 100)–ESs, sphere model, $n = 100$ and (15, 100)–ESs, scaling test, $n = 100$

In both cases, the performance of the new algorithm can be found in between the best and the worst settings for both problems. Especially when dealing with a fitness function not being symmetric in all coordinate directions, an algorithm which has the capability of learning the appropriate number of stepsizes during the search performs much better compared to the standard ES with the number of stepsizes fixed to one. But can the divergence observed in Figure 2 and 3 be avoided now? As Figure 5 and 6 indicate, this problem still remains to be solved. On the other hand, the ES with a variable number of stepsizes converges faster than both alternatives for some settings of the recombination operator. Although the scaling test function is said to need n stepsizes for a fast convergence, this only holds for those four settings involving intermediate recombination: ii- (441), iI- (451), Ii- (541) and II- (551). The same four settings perform best optimizing the sphere model with n stepsizes. They even converge faster than some of the 'appropriate' variants with only one stepsize for all object variables x_i.

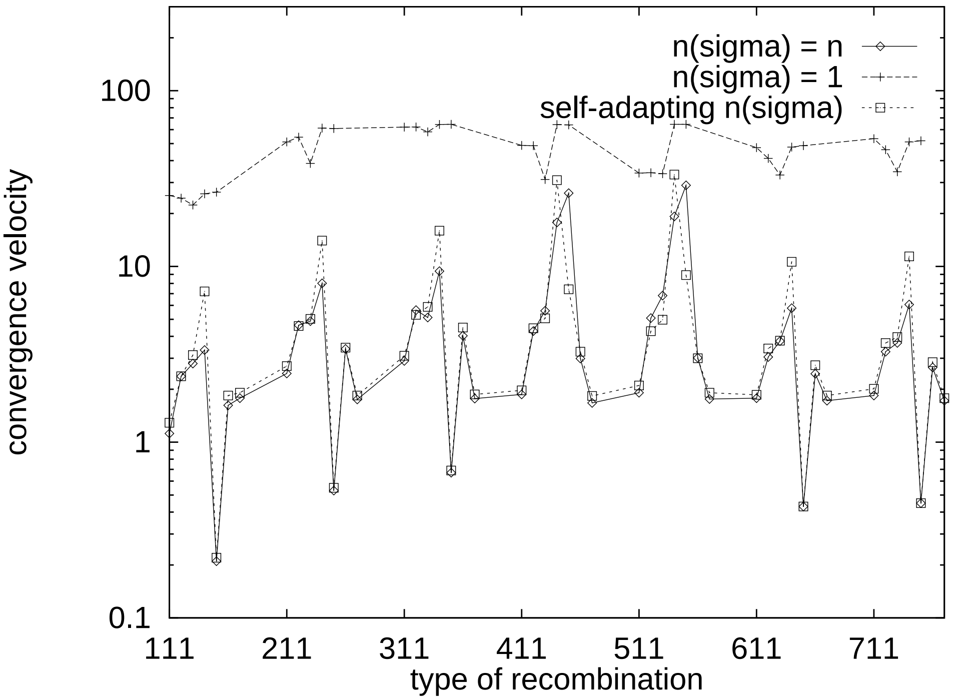

Fig. 5: (15, 100)–ES, sphere model, $n = 100$

The results depicted in Figure 5 and 6 lead the way for further research. Adapting the number of stepsizes works in a black box–type of situation, but the additional adaptability of the recombination operator seems to be even more important.

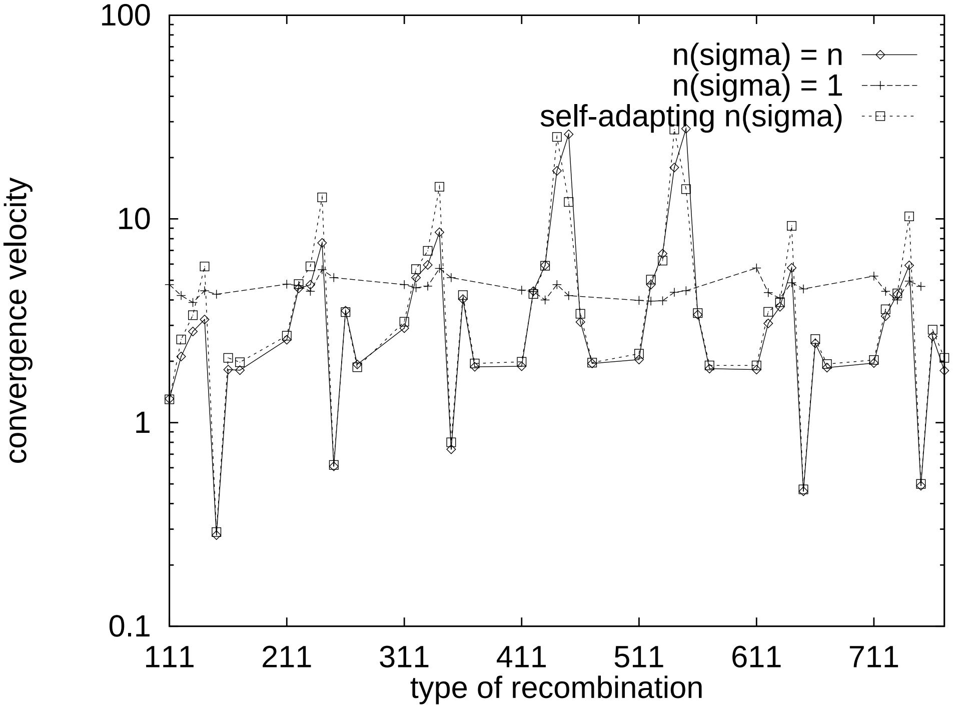

Fig. 6: (15, 100)–ES, scaling test, $n = 100$

As far as GAs are concerned, a similar study on control parameters has been done. Schaffer et al. [5] found out that the parameters (population size, crossover rate, mutation rate) which were recommended by Ken De Jong [3] differed significantly from their results.

4. Outlook

Provided with some starting values, the evolution strategy itself ought to change its parameters for a faster and more reliable convergence. As far as the number of stepsizes is concerned first results presented in this paper indicate that this concept is reasonable. Further research will show whether other exogenous parameters can self–adapt, too, like e. g. the recombination operator which seems to be crucial for both convergence velocity and reliability. The 'learning rates' τ' and τ which determine the adaptability of the stepsizes could also become part of an individual's genetic material. Furthermore, thinking of a multi–population concept, it should be possible to breed good, if not optimal settings of the ES for a given problem. First results have already shown that this idea of an evolution of strategies is feasible.

References

[1] T. Bäck, F. Hoffmeister, and H.-P. Schwefel, "A survey of evolution strategies", in *Proceedings of the Fourth International Conference on Genetic Algorithms*, (R. K. Belew and L. B. Booker, eds.), pp. 2–9, Morgan Kaufmann, San Mateo, CA, 1991.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, Massachusetts: Addison–Wesley, 1989.

[3] K. A. D. Jong, *An analysis of the behaviour of a class of genetic adaptive systems.* PhD thesis, University of Michigan, 1975. Diss. Abstr. Int. 36(10), 5140B, University Microfilms No. 76–9381.

[4] I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Stuttgart: Frommann–Holzboog, 1973.

[5] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das, "A study of control parameters affecting online performance of genetic algorithms for function optimization", in *Proceedings of the 3rd International Conference on Genetic Algorithms*, (J. D. Schaffer, ed.), pp. 51–60, Morgan Kaufmann Publishers, San Mateo, CA, 1989.

[6] H.-P. Schwefel, *Numerische Optimierung von Computer–Modellen mittels der Evolutionsstrategie.* Basel: Birkhäuser, 1977.