Stopping Criteria for a Constrained Single-Objective Particle Swarm Optimization Algorithm

Karin Zielinski and Rainer Laur
Institute for Electromagnetic Theory and Microelectronics, University of Bremen, Germany
{zielinski,rlaur}@item.uni-bremen.de

Keywords: constrained single-objective optimization, evolutionary algorithms, particle swarm optimization, stopping criteria

Received: November 3, 2006

When using optimization algorithms the goal is usually clear: The global optimum should be found. However, in general it is not clear when this goal is achieved, especially if real-world problems are optimized for which no knowledge about the global optimum is available. Therefore, it is not easy to decide when the execution of an optimization algorithm should be terminated. Although different mechanisms can be used for the detection of an appropriate time for ending an optimization run, only two of them are frequently used in the literature. Unfortunately, both methods have disadvantages, particularly for the optimization of real-world problems. Because especially for practical applications it is important when an optimization algorithm is terminated as they usually contain computationally expensive objective functions, the performance of several stopping criteria that react adaptively to the state of an optimization run is evaluated for a Particle Swarm Optimization algorithm in this work. The examination is done on the basis of a constrained single-objective power allocation problem. Suggestions from former work concerning stopping criteria for unconstrained optimization are verified and comparisons with results for Differential Evolution are made.

Povzetek: Ovrednoteni so ustavitveni kriteriji za optimiranje z roji delcev (angl. particle swarm optimization) in rezultati primerjani z rezultati algoritma diferencialne evolucije.

1 Introduction

Evolutionary algorithms (EAs) are a class of population-based stochastic optimization algorithms that incorporate mechanisms from evolution for optimization processes. The most famous representatives from this class are possibly Genetic Algorithms [5] but in the last years also e.g. Particle Swarm Optimization (PSO) [7] and Differential Evolution (DE) [9] had a lot of success.

For theoretical aspects of evolutionary algorithms stopping criteria are usually not important. However, for practical applications the choice of stopping criteria can significantly influence the duration of an optimization run. Due to different stopping criteria an optimization run might be terminated before the population has converged, or computational resources might be wasted because the optimization run is terminated late. Real-world problems mostly contain computationally expensive objective functions that may result in optimization runs that take several days, thus wasting of computational resources has to be prevented.

In the literature mostly two stopping criteria are applied in single-objective optimization: Either an error measure in dependence on the known optimum is used or the number of function evaluations is limited to fe_{max}. These criteria are perfectly suitable for e.g. comparing the performance of different algorithms but for solving real-world problems there are some drawbacks. The first mentioned method has the disadvantage that the optimum has to be known, so it is generally not applicable to real-world problems because the optimum is usually not known a priori. The second method is highly dependent on the objective function. Because generally no correlation can be seen between an optimization problem and the required number of function evaluations, fe_{max} has to be determined by trial-and-error methods usually. Evolutionary algorithms include randomness in the optimization process, thus the number of function evaluations that is needed for convergence is subject to fluctuations, so a safety margin for fe_{max} is needed. The fluctuations can be significant as can be seen in [17] where a test suite of 24 functions has been examined, and the standard deviation of function evaluations for reaching a predefined error measure was up to 180,000. If a real-world problem with an unknown optimum would result in a similar standard deviation, it would be difficult to choose fe_{max}.

As a result, it would be better to use stopping criteria that consider knowledge from the state of the optimization run. The time of termination would be determined adaptively, so function evaluations could be saved.

Several stopping criteria are reviewed in [19] and [20] that are sensitive to the state of the optimization run by observing the improvement, movement or distribution of the population members. In [19] stopping criteria are tested for unconstrained single-objective optimization using Particle Swarm Optimization and Differential Evolution, while in

[20] the criteria have been adapted for constrained single-objective problems using DE because real-world problems normally include constraints. In this work it will be examined if the suggestions regarding stopping criteria for PSO from [19] hold for the constrained real-world problem of optimizing a power allocation scheme. Additionally, a comparison with the results for DE in [20] will be done.

This work is organized as follows: In Section 2 related work is discussed. In Section 3 the Particle Swarm Optimization algorithm is described and Section 4 provides a short introduction to Differential Evolution. In Section 5 the stopping criteria that are used in this work are reviewed. In Section 6 results are shown and Section 7 closes with conclusions.

2 Related Work

Every optimization algorithm includes a stopping rule but there are only few works concentrating explicitly on stopping criteria. In [16] convergence of a Particle Swarm Optimization algorithm is detected by computing a maximum swarm radius, by doing a cluster analysis or by calculating the rate of change in the objective function. Most stopping criteria are applicable not only to PSO but also to other population-based optimization algorithms, e.g. in [1] the difference between maximum and minimum objective function value is used as stopping criterion for a Differential Evolution algorithm. In [13] not only termination criteria for evolutionary algorithms but also for other optimization algorithms are discussed. Often criteria similar to the ones used in the work are also applied in hybrid algorithms to determine the moment when global search should be replaced by local search [4, 6, 15].

3 Particle Swarm Optimization

Particle Swarm Optimization is derived from the behavior of social groups like bird flocks or fish swarms. Although the "survival of the fittest" principle is not used in PSO, it is usually considered as an evolutionary algorithm. A thorough discussion of this topic can be found in [7]. Like in this work, PSO is mostly used for the optimization of continuous functions.

Optimization is achieved by giving each individual in the search space a memory for its previous successes, information about successes of a social group and providing a way to incorporate this knowledge into the movement of the individual. Therefore, each individual (called particle) is characterized by its position $\vec{x}_i$, its velocity $\vec{v}_i$, its personal best position $\vec{p}_i$ and its neighborhood best position $\vec{p}_g$. Several neighborhood topologies have been developed [10]. In this work the *von-Neumann* topology is used as it showed promising results in the literature, e.g. in [8].

The dynamic behavior of PSO is generated by the update equations for velocity and position of the particles:

$$\begin{aligned}\vec{v}_i(t+1) &= w \cdot \vec{v}_i(t) \\ &\quad +c_1 r_1[\vec{p}_i(t) - \vec{x}_i(t)] \\ &\quad +c_2 r_2[\vec{p}_g(t) - \vec{x}_i(t)]\end{aligned} \tag{1}$$

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \tag{2}$$

Due to these equations the particles are drawn towards their personal best position and their neighborhood best position, and furthermore the velocity of the previous iteration is kept weighted with the inertia weight w. Other parameters are c_1 and c_2 which influence the impact of the cognitive and social component, respectively. To add a stochastic element to the movement of the particles, the numbers r_1 and r_2 are chosen randomly from the interval [0,1] in each iteration. Further parameters of PSO are the population size NP and the maximum velocity V_{max} that is used for preventing oscillations with increasing magnitude [7].

The control parameter settings for this examination are derived from a parameter study using the same optimization problem (yet unpublished): $w = 0.6$, $c_1 = 0.4$, $c_2 = 1.4$, $NP = 64$, $V_{max} = \frac{1}{2}(X_{max} - X_{min})$.

Constraint-handling is done by modifying the replacement procedure for personal and neighborhood best positions [11]. In unconstrained single-objective optimization a personal or neighborhood best position is replaced if the current position has a lower objective function value (for minimization problems as in this work). For constrained single-objective optimization this rule is altered so that in a comparison of two solutions $\vec{a}$ and $\vec{b}$, $\vec{a}$ is preferred if

- both vectors are feasible and $\vec{a}$ has a better objective function value or
- both vectors are infeasible and $\vec{a}$ has the lower sum of constraint violation or
- $\vec{a}$ is feasible and $\vec{b}$ is infeasible

where feasibility means that all constraints are satisfied. In contrast to several other constraint-handling techniques, no additional parameters are needed for this method [2]. For unconstrained problems the modified PSO algorithm is exactly the same as the original PSO.

4 Differential Evolution

The main characteristic of Differential Evolution is an adaptive scaling of step sizes that results in fast convergence behavior. Using DE the population members are evolved from one generation to the next by applying the operators mutation, recombination and selection. The first two operators generate new vectors by linearly combining several population members and afterwards exchanging some vector components. The third operator decides based on objective function values and constraint violation which vectors will be kept for the next generation. Because no deterioration with regard to the objective function value

is possible, the DE selection scheme is called greedy [14]. More specific information about the here mentioned DE algorithm can be found in [20].

5 Stopping Criteria

Stopping criteria are needed to terminate the execution of optimization algorithms. In contrast to using a maximum number of function evaluations as a stopping condition, other criteria have the advantage of reacting adaptively to the state of the optimization run, thus function evaluations can be saved. Unfortunately, it seems to be impossible to define a stopping criterion without introducing one or more parameters. The parameter settings generally depend on the given optimization problem. However, it should be investigated if there are stopping criteria for which the parameter settings are robust to changes or if parameters can be set depending on certain aspects of the problem. It is assumed that the general behavior of different optimization problems to stopping criteria is similar. It should be kept in mind that limiting the number of function evaluations as a stopping criterion also incorporates the choice of a problem-dependent parameter fe_{max}. Hence, it is favorable to examine other possibilities for stopping that contain the advantage of reacting adaptively to the state of the optimization run.

In the following the stopping criteria that incorporate information about the state of the optimization run are reviewed shortly. Note that there is a change compared to [19]: Instead of using the current positions $\vec{x}_i$ for the calculation of stopping conditions, the personal best positions $\vec{p}_i$ are used here. The reason is that the current positions have many fluctuations whereas the development of the personal best positions is more smooth, so decisions about termination of an optimization run should be easier.

Improvement-based criteria terminate an optimization run if only small improvement is made because usually in the beginning of an optimization run large improvements are achieved whereas in later stages the improvement becomes small. Three different conditions are used here:

- *ImpBest*: The improvement of the best objective function value is monitored. If it falls below a given threshold t for a number of generations g, the optimization run is terminated.
- *ImpAv*: Similar to *ImpBest*, but instead of observing the best objective function value, the average value computed from the whole population is checked.
- *NoAcc*: It is observed if any new $\vec{p}_i$ are accepted in a specified number of generations g. For DE this criterion is slightly different because in DE there are no personal best positions (instead, the acceptance of new population members is considered).

For *movement-based criteria* not the improvement but the movement of individuals is regarded. Two variants of movement-based criteria are considered that differ in the regarded space:

- *MovObj*: The movement of the individuals with respect to their objective function value (objective space) is examined if it is below a threshold t for a number of generations g. *MovObj* is different from *ImpAv* only if the regarded algorithm allows deterioration of the individuals' objective function value. This is the case for PSO in contrast to DE, but as $\vec{p}_i$ are considered here instead of $\vec{x}_i$, *MovObj* = *ImpAv* holds in this case also. Therefore, this criterion is not regarded further in this work.
- *MovPar*: The movement with respect to positions (parameter space) is checked if it is below a threshold t for a number of generations g.

The *distribution-based criteria* consider the diversity in the population. If the diversity is low, the individuals are close to each other, so it is assumed that convergence has been obtained.

- *StdDev*: It is checked if the standard deviation of positions is below a threshold m.
- *MaxDist*: The distance from every population member to the best individual is observed. The optimization run is stopped if the maximum distance is below a threshold m.
- *MaxDistQuick*: *MaxDistQuick* is a generalization of *MaxDist*. Instead of using the whole population for the computation of the maximum distance to the best population member, a quicksort algorithm is employed for sorting the individuals due to their objective function value, and only the best $p\%$ of the individuals are regarded. The background for this criterion is that there are optimization runs where most of the population has converged to the optimum but because of the remaining individuals which are still searching, the optimization run is not stopped although they do not contribute any new information. Using *MaxDistQuick* an optimization run can be stopped earlier than using *MaxDist*, so wasting of computational resources is avoided. However, the percentage p must not be set too low for a reliable detection of convergence.
- *Diff*: The difference between best and worst objective function value is checked if it is below a threshold d. A further demand is that at least $p\%$ of the individuals are feasible because otherwise *Diff* could lead to undesired results if e.g. only two individuals are feasible and they are close to each other incidentally. In contrast to the previous three criteria that are used in parameter space, *Diff* considers objective space.

Because functions have different features it may be beneficial to couple several criteria. Up to now two *combined criteria* have been regarded:

– *ComCrit*: This criterion is a combination of *ImpAv* and *MaxDist*. Only if the condition of *ImpAv* is fulfilled, *MaxDist* is checked.

– *Diff_MaxDistQuick*: *Diff* is a criterion that is rather easy to check, but it fails with flat surfaces. Therefore, if its condition has been fulfilled, the *MaxDistQuick* criterion is checked afterwards.

6 Results

As a basis for the examination a real-world problem was used that consists of optimizing a power allocation scheme for a Code Division Multiple Access (CDMA) system [20]. The overall power is minimized considering the powers of 16 individual users as parameters. Because multiple users send data simultaneously in a CDMA system, multi-user interference degrades the system performance. The application of a parallel interference cancellation technique allows estimation of the multi-user interference, so it can be subtracted from the received signal before detection, resulting in improvement of the system performance. The convergence of the parallel interference cancellation technique has to be incorporated in the optimization problem as a constraint.

In the following results are shown sorted according to the type of stopping criterion. The global optimum is considered to be reached if an objective function value of $f(x) \leq 18.5$ has been found [20]. As performance measures the convergence rate and the success performance (mean number of function evaluations weighed with the total number of runs divided by the number of successful runs) are given. A high convergence rate and a small success performance indicate good performance. To allow easy comparisons, figures showing success performances are scaled to 20,000. A maximum number of generations $G_{max} = 1000$ is used to terminate the algorithm if the examined stopping criteria do not lead to termination in appropriate time. If a run is not stopped before G_{max} is reached, the run is considered as unsuccessful.

6.1 Improvement- and Movement-Based Criteria

Because *ImpAv*, *ImpBest* and *MovPar* rely on similar mechanisms, the convergence rate and success performance of these criteria are displayed together. Considering the convergence rate, almost no dependence on the number of generations g is observable (Figure 1(a)). For decreasing values of the improvement threshold t generally the convergence rate increases, except for *MovPar* that was not able to terminate several runs before reaching G_{max} for small settings of t.

The success performance of *ImpAv*, *ImpBest* and *MovPar* is slightly increasing with growing g (see Figure 1(b)). The results regarding t are similar for *ImpAv* and *ImpBest*: For high settings of t the success performance is large because of the small convergence rate. After a strong decrease the success performance increases again for smaller values of t because of the growing average number of function evaluations for convergence.

The smallest success performance of *MovPar* is in the same range as for *ImpAv* and *ImpBest*. The difference in the average number of function evaluations for different settings of t is larger for *MovPar* than for *ImpAv* or *ImpBest*, thus the success performance grows quickly for decreasing t. As a result the success performance is better for $t = 10^{-2}$ than for $t = 10^{-4}$ although the convergence rate of $t = 10^{-2}$ is worse.

The success performance of *ImpAv* and *MovPar* has similar characteristics as for DE in [20]. For *ImpBest* the results are different: The success performance for $g = 5$ is considerably better for PSO. Furthermore, the success performance is dependent on t and almost independent from g whereas for DE it depends more on g than on t. The reason for the different results is not clear yet.

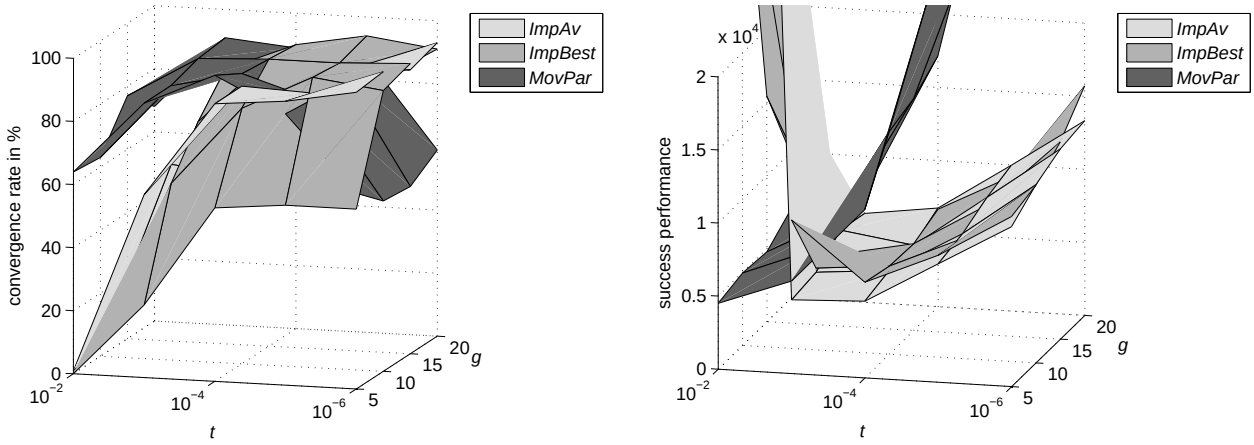
The results for *ImpAv* and *ImpBest* are considerably better here than in [19] for unconstrained single-objective problems. For *ImpAv* the reason might be that the personal best positions are regarded here instead of the current positions, but criterion *ImpBest* did not change because only the global best result is regarded. In contrast, for *MovPar* the results are worse, but it has to be kept in mind that the results are slightly dependent on the setting of G_{max} because it influences the convergence rate.

Unfortunately, suitable parameter settings for *ImpAv* and *ImpBest* cannot be derived from knowledge about the optimization problem. Besides, it is indicated in [19] that problems arise for functions with a flat surface, but it is usually not known in advance if a function possesses this property. Therefore, it will be necessary to do examinations on parameter settings for the application of these stopping criteria. Based on the examined problem parameter settings of $g \approx 10 \ldots 15$ and $t \approx 10^{-5} \ldots 10^{-4}$ are recommended. However, these settings are dependent on the optimization problem and the desired accuracy. It has to be noted also that these criteria may not be as reliable as others because improvement often occurs irregularly in evolutionary algorithms.

Criterion *NoAcc* showed good results for DE in [20] but not a single run could be terminated before reaching G_{max} for PSO. Apparently, the personal best positions improve too often to allow a stopping criterion like *NoAcc*.

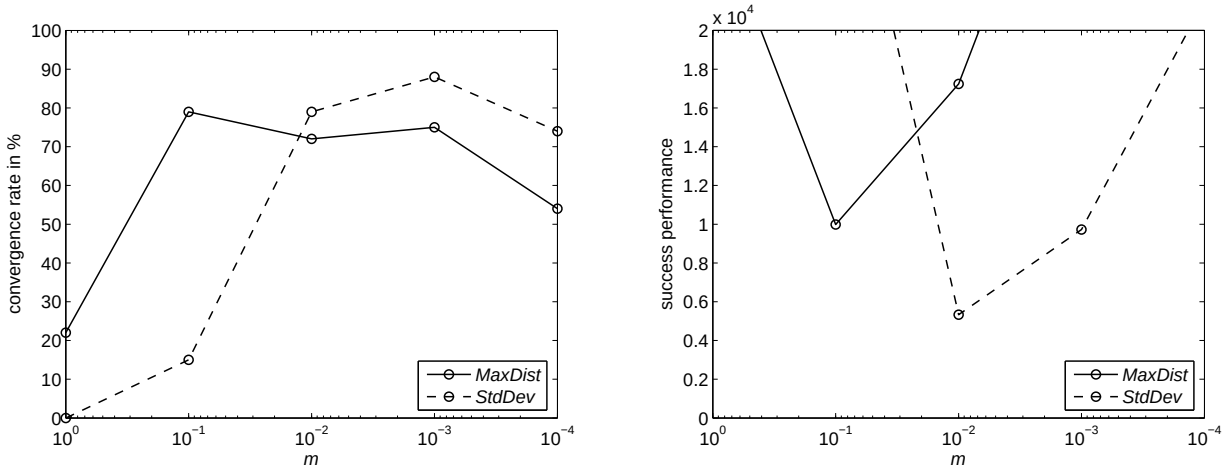
6.2 Distribution-Based Criteria

For *MaxDist* the convergence rate does not get above 80% because of runs that could not be terminated before reaching G_{max}. The results for *StdDev* are shifted in contrast to *MaxDist* and higher convergence rates are reached (Figure 2(a)). Furthermore, *StdDev* yields a lower minimum success performance than *MaxDist* (Figure 2(b)). For both criteria the performance is highly dependent on the setting



(a) Convergence rate (b) Success performance

Figure 1: Results for criteria *ImpAv*, *ImpBest* and *MovPar*



(a) Convergence rate (b) Success performance

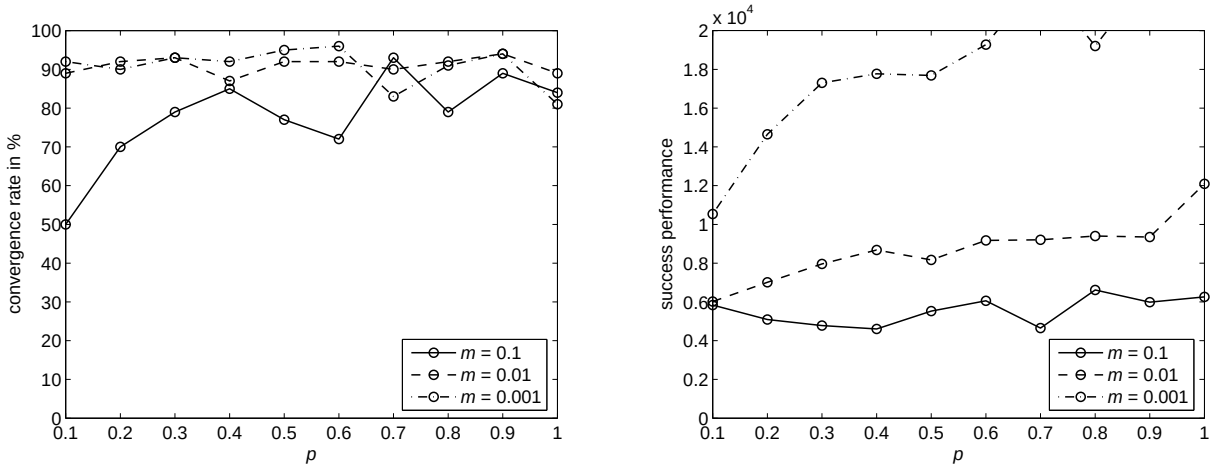
Figure 2: Results for criteria *MaxDist* and *StdDev*

of m. However, it is connected to the desired accuracy of the result. Similar effects have been found in [20] for DE. The same settings of parameter m yield the lowest success performances for *MaxDist* and *StdDev* for PSO as for DE, respectively.

The convergence rate and success performance of *MaxDistQuick* is given for $10^{-3} \leq m \leq 10^{-1}$ in Figures 3(a) and 3(b). Other parameter settings are omitted because the success performance was above 20,000. The convergence rate is fluctuating for $m = 0.1$ with different settings of p, indicating that the performance is not robust for this parameter setting. For $m = \{10^{-2}, 10^{-3}\}$ and varying p the convergence rate is approximately constant but the success performance rises with increasing p. Hence, a similar result is obtained as in [19]: Because less function evaluations are needed for convergence if smaller values of p are used and the convergence probability is not compromised, it is recommended to use e.g. $0.3 \leq p \leq 0.5$. In spite of the increased computational effort for the incorporated quicksort algorithm [18], *MaxDistQuick* is considered to be superior to *MaxDist* and *StdDev* for PSO, particularly because the increased computational effort is usually negligible when compared to computationally expensive objective function evaluations of real-world problems. For future work also a similar generalized criterion based on standard deviation instead of maximum distance should be evaluated.

For DE the success performance depends less on p [19, 20], so *MaxDistQuick* does not have advantages over *MaxDist* for DE. This behavior is supposed to be connected with the greediness of the DE selection scheme.

It may be confusing that the success performance for *MaxDistQuick* with $p = 1$ is not equal to the results of *MaxDist*. The reason is that the success performance is sensitive to even small changes in the number of success-



(a) Convergence rate

(b) Success performance

Figure 3: Results for criterion *MaxDistQuick*

ful runs. If the average number of function evaluations is examined, the results from *MaxDistQuick* with $p = 1$ and *MaxDist* are similar (not shown here).

For criterion *Diff* no definite trend can be observed regarding the demanded percentage p of feasible individuals in the population (Figures 4(a) and 4(b)) which is assumed to be due to the fact that all individuals get feasible quite fast here. Similar results were found for DE in [20]. As expected, the success performance depends on the difference threshold d. Like parameter m of the other distribution-based criteria, the setting of d is connected with the desired accuracy of the result. The highest convergence rate is achieved with $d = 10^{-2}$ but although $d = 10^{-1}$ results in a worse convergence rate, the success performance is better.

Criterion *Diff* is advantageous in contrast to the distribution-based criteria in parameter space if several parameter combinations yield the same objective function value. In this case the distribution-based criteria in parameter space may waste computational resources while the algorithm tries to converge to one point in the search space, with no or only little improvement of the objective function value. However, *Diff* is likely to produce bad results for functions with a flat surface [19].

6.3 Combined Criteria

The convergence rate and success performance for both combined criteria are given for $m \geq 10^{-2}$ because smaller values of m lead to success performances larger than 20,000 (Figures 5(a), 5(b), 6(a) and 6(b)). The results are different than for DE as the success performance increases less with decreasing value of m. Especially for *Diff_MaxDistQuick* the results are almost independent from m. However, a strong dependence on d can be seen, in particular for the success performance.

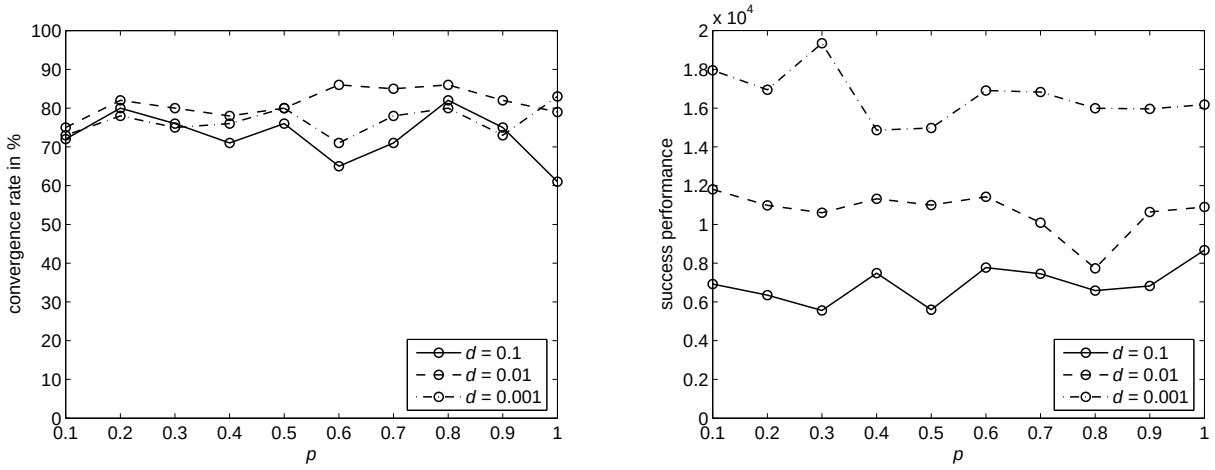
For the combined criteria generally more parameters have to be set than for the individual criteria and furthermore the dependence of parameter settings on the desired accuracy of the results cannot be seen anymore, so in general it might be easier to use the individual criteria.

6.4 Summary

Although the improvement-based criteria *ImpAv* and *ImpBest* yielded good results in this work, they are considered as rather unreliable because generally improvement occurs irregularly in evolutionary algorithms. To prevent early termination, parameter g must not be chosen too low when using these criteria. The movement-based criterion *MovPar* has similar problems. The third improvement-based criterion *NoAcc* was not able to stop a single optimization run during the given maximum number of generations, so it is classified as unsuitable for PSO although it showed a good performance for DE in [20].

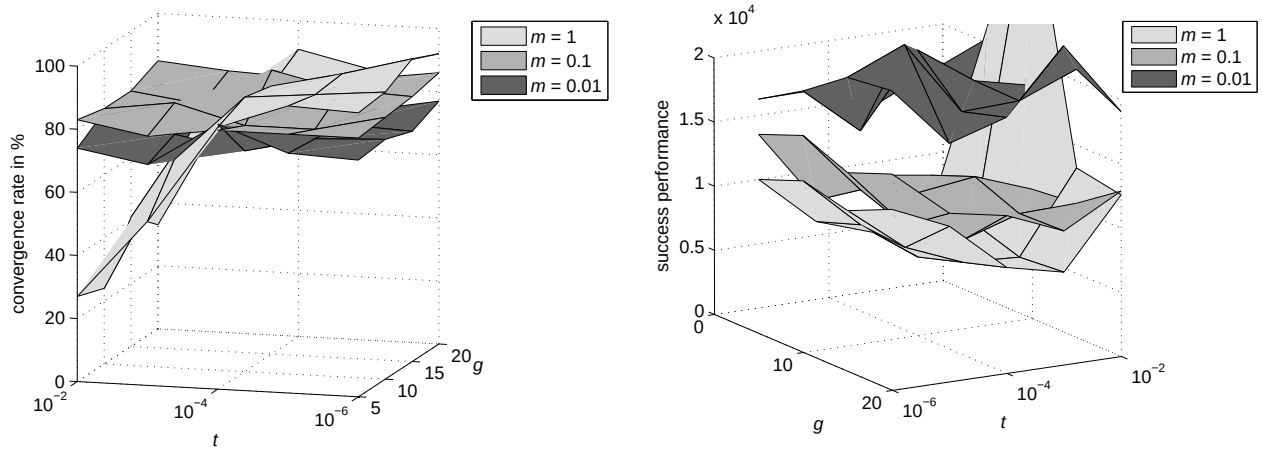
Based on the considered optimization problem as well as results from [19] it can be concluded that it is beneficial to use the generalization *MaxDistQuick* instead of *MaxDist*. Because *StdDev* performed better than *MaxDist*, a generalization of *StdDev* should be examined in future work. In general the distribution-based criteria in parameter space are classified as reliable means for detecting convergence. The distribution-based criterion in objective space (*Diff*) is also considered to be a reliable criterion with the exception of optimization problems that contain objective functions with a flat surface.

As the combined criteria are combinations of other criteria, they generally incorporate more parameters that have to be adjusted. So far no advantage of combined criteria could be found that would compensate this drawback, so it is recommended to use the individual criteria.



(a) Convergence rate (b) Success performance

Figure 4: Results for criterion *Diff*



(a) Convergence rate (b) Success performance

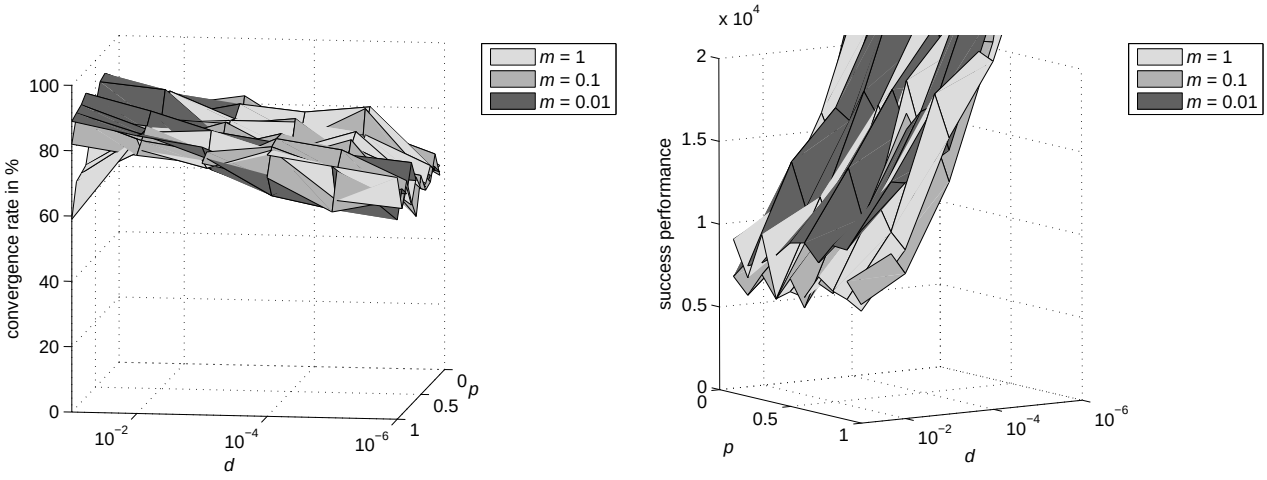
Figure 5: Results for criterion *ComCrit*

7 Conclusions

In this work stopping criteria were studied that react adaptively to the state of an optimization run based on improvement, movement or the distribution of individuals. In contrast to other examinations, not the current positions but the personal best positions were used for the calculations. It was shown that the stopping criteria can be used for constrained problems using PSO. A similar behavior as for DE could be found for several stopping criteria. It would be interesting to make comparisons with other evolutionary algorithms in future work.

Although parameter settings have to be determined in dependence on the used optimization problem, general statements could be derived. It was not possible to determine one criterion that will be best for all problems, but because of their adaptive nature generally improved performance for real-world problems is expected in contrast to termination after a limited number of function evaluations.

For multi-objective optimization the definition of appropriate stopping criteria is even more important because real-world problems usually contain multiple objectives. It will be also even more challenging because usually the population will not converge to one point in the search space but to the Pareto-optimal front, thus using error measures is difficult. One possibility would be to monitor performance measures like hypervolume [3] and calculate e.g. improvement. Another approach from literature is based on observing the development of crowding distances [12]. As only little work is done in this area so far, it is an interesting field of research for future work.



(a) Convergence rate

(b) Success performance

Figure 6: Results for criterion *Diff_MaxDistQuick*

References

[1] B. V. Babu and Rakesh Angira. New Strategies of Differential Evolution for Optimization of Extraction Process. In *Proceedings of International Symposium & 56th Annual Session of IIChE (CHEMCON 2003)*, Bhubaneswar, India, 2003.

[2] Carlos A. Coello Coello. Theoretical and Numerical Constraint-Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287, 2002.

[3] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, 2001.

[4] Felipe P. Espinoza. *A Self-Adaptive Hybrid Genetic Algorithm for Optimal Groundwater Remediation Design*. PhD thesis, University of Illinois, 2003.

[5] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[6] Wilfried Jakob. *Eine neue Methode zur Erhöhung der Leistungsfähigkeit Evolutionärer Algorithmen durch die Integration lokaler Suchverfahren*. PhD thesis, Universität Karlsruhe, 2004.

[7] James Kennedy and Russell C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, San Francisco, 2001.

[8] James Kennedy and Rui Mendes. Population Structure and Particle Swarm Performance. In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Garry Greenwood, Hitoshi Iba, Paul Marrow, and Mark Shackleton, editors, *Proceedings of the Congress on Evolutionary Computation (CEC 2002)*, pages 1671–1676, Honolulu, HI, USA, 2002.

[9] Jouni Lampinen and Rainer Storn. Differential Evolution. In Godfrey C. Onwubolu and B.V. Babu, editors, *New Optimization Techniques in Engineering*, pages 123–166. Springer-Verlag, Berlin Heidelberg, 2004.

[10] Rui Mendes, James Kennedy, and José Neves. The Fully Informed Particle Swarm: Simpler, Maybe Better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210, 2004.

[11] Gregorio Toscano Pulido and Carlos A. Coello Coello. A Constraint-Handling Mechanism for Particle Swarm Optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC 2004)*, volume 2, pages 1396–1403, Portland, OR, USA, 2004.

[12] Olga Rudenko and Marc Schoenauer. A Steady Performance Stopping Criterion for Pareto-based Evolutionary Algorithms. In *Proceedings of the 6th International Multi-Objective Programming and Goal Programming Conference*, Hammamet, Tunisia, 2004.

[13] Hans-Paul Schwefel. *Evolution and Optimum Seeking*. John Wiley and Sons, 1995.

[14] Rainer Storn and Kenneth Price. Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[15] Michael Syrjakow and Helena Szczerbicka. Combination of Direct Global and Local Optimization Methods. In *Proceedings of the IEEE International Conference on Evolutionary Computing (ICEC 95)*, pages 326–333, Perth, WA, Australia, 1995.

[16] Frans van den Bergh. *An Analysis of Particle Swarm Optimizers*. PhD thesis, University of Pretoria, 2001.

[17] Karin Zielinski and Rainer Laur. Constrained Single-Objective Optimization Using Particle Swarm Optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1550–1557, Vancouver, BC, Canada, 2006.

[18] Karin Zielinski, Dagmar Peters, and Rainer Laur. Run Time Analysis Regarding Stopping Criteria for Differential Evolution and Particle Swarm Optimization. In *Proceedings of the 1st International Conference on Experiments/Process/System Modelling/Simulation/Optimization*, Athens, Greece, 2005.

[19] Karin Zielinski, Dagmar Peters, and Rainer Laur. Stopping Criteria for Single-Objective Optimization. In *Proceedings of the Third International Conference on Computational Intelligence, Robotics and Autonomous Systems*, Singapore, 2005.

[20] Karin Zielinski, Petra Weitkemper, Rainer Laur, and Karl-Dirk Kammeyer. Examination of Stopping Criteria for Differential Evolution based on a Power Allocation Problem. In *Proceedings of the 10th International Conference on Optimization of Electrical and Electronic Equipment*, volume 3, pages 149–156, Braşov, Romania, 2006.