

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/221007645>

Crossover and the different faces of differential evolution searches

Conference Paper · July 2010

DOI: 10.1109/CEC.2010.5586184 · Source: DBLP

CITATIONS

14

READS

87

1 author:



James Montgomery

University of Tasmania

44 PUBLICATIONS 507 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Towards Exploration-only, Exploitation-only Hybrid Search Techniques [View project](#)

Crossover and the Different Faces of Differential Evolution Searches

James Montgomery

Abstract—Common explanations of DE's search behaviour as its crossover rate Cr is varied focus on the directionality of the search, as low values make moves aligned with a small number of axes while high values search at angles to the axes. While the direction of search is important, an analysis of moves generated by mutating differing numbers of dimensions suggests that the probability of making a successful move is more strongly related to the move's magnitude than to the number of dimensions in which it occurs. Low Cr moves are generally much smaller than those generated with high values, and more likely to succeed, but moves in many dimensions can produce greater improvements in solution quality. Although DE behaves differently at low and high Cr , both extremes can produce effective searches. Results suggest this is because low Cr searches make frequent, small improvements to all population members while high Cr searches produce less frequent, large improvements, followed by contraction of the population and a resultant reduction in move size. The interaction of F and population size with these different modes of search is investigated and recommendations made to achieve good results with both.

I. INTRODUCTION

Differential evolution (DE) [1] is a population-based search technique used successfully for optimisation in numerous continuous domains [2]. A key parameter that controls its search behaviour and, consequently, performance is its crossover rate (Cr). Due to the mechanisms that control the generation of new solutions (detailed below for those unfamiliar with the algorithm) low values of Cr produce exploratory moves parallel to a small number of search space axes, while larger values produce moves at angles to most or all axes. Thus low values have been favoured when solving separable problems whereas values around 0.9 (but not 1) are preferred for non-separable problems [3]. While these respective modes of search are indeed well suited to those particular kinds of problem [3], [4], explanations of Cr 's role in DE's performance that focus almost exclusively on the *directionality* of the search miss a key difference between them: the magnitude of the moves made.

This paper investigates properties of the moves DE makes for different values of Cr and their effects on its search behaviour. This analysis provides insights into why low and high values can be effective. The interaction of the other key parameters of vector scale F and population size Np is also examined. First, an analysis is made of the characteristics of exploratory moves that might be made using a combination of vector addition and crossover. This analysis informs a subsequent study of DE's performance on different search landscapes as Cr is varied (Section III). While Np has

differing effects on the search when Cr is low or high, the scale factor F has the greatest impact when Cr is high. The interaction of these two parameters is discussed in Section IV. The explanation of how and why the control parameters affect DE's behaviour will assist in its future application and in the development of improved adaptive algorithms.

A. Differential Evolution

DE is a generational evolutionary algorithm in which, at each iteration, every population member is considered as a *target* for replacement by a newly generated point in solution space. The most common, and frequently effective [5], variant is designated DE/rand/1/*, which is the subject of this work. DE/rand/1/* generates new solutions by adding the weighted difference between two randomly selected population members to a third population member (not the *target*). The * may be either *bin* for a uniform crossover, where the probability of mutating a component follows a binomial distribution, or *exp*, where a sequence of vector components is taken, the length of which follows an inverse exponential distribution. Although the number of components mutated differs between *bin* and *exp* for the same value of Cr , when the actual probability of mutation is equivalent there are negligible differences between the performance of the two [3], so only the *bin* variant is considered here. Let $S = \{x_1, x_2, \dots, x_{|S|}\}$ be the population of solutions. In each iteration, each solution in S is considered as a *target* for replacement by a new solution; denote the current target by x_i . A new point v_i is generated according to

$$v_i = x_{r1} + F \cdot (x_{r2} - x_{r3}) \quad (1)$$

where x_{r1} (also referred to as the *base*), x_{r2} and x_{r3} are distinct, randomly selected solutions from $S \setminus \{x_i\}$ and F is the scaling factor, typically in $(0, 1]$ although larger values are also possible. Uniform crossover is performed on v_i , controlled by the parameter $Cr \in [0, 1]$, to produce the candidate solution u_i , according to

$$u_i^j = \begin{cases} v_i^j & \text{if } R_j \leq Cr \text{ or } j = I_i, \\ x_i^j & \text{if } R_j > Cr \text{ and } j \neq I_i, \end{cases} \quad (2)$$

where $R_j \in [0, 1)$ is a uniform random number and I_i is the randomly selected index of a component that must be mutated, which ensures that $u_i \neq x_i$. The *target* is replaced if the new solution is as good or better.

Small values of Cr result in exploratory moves parallel to a small number of axes of the search space, while large values of Cr produce moves at angles to the search space's axes. Consequently, the general consensus, supported by some empirical studies [3], [4], is that small Cr is useful when

James Montgomery is with the Complex Intelligent Systems Laboratory, Faculty of Information & Communication Technologies, Swinburne University of Technology, Melbourne, Australia (phone: +61 9214 5735; email: jmontgomery@swin.edu.au).

solving separable problems while large Cr is more useful when solving non-separable problems.

B. Previous Studies of Cr 's Effects on Performance

Despite the important role that crossover plays in the DE algorithm there are relatively few studies that have examined the algorithm's performance as Cr is varied. Price, Storn and Lampinen [2] performed a sensitivity analysis of Cr and F on a number of rotated problems. The results suggest that on rotated, originally separable problems, both Cr and F must be above 0.5, while on irregular landscapes all values of Cr could succeed provided $F > 0.5$. Rönkkönen, Kukkonen and Price [4] examined the relationship between Cr and DE's performance on different problems, concluding that $Cr \leq 0.2$ was appropriate for separable problems and $Cr > 0.9$ was best for non-separable problems. Mezura-Montes, Velázquez-Reyes and Coello Coello [5] performed a sensitivity analysis to pick the "best" setting for Cr for various problems. In many instances a low value of $Cr = 0.1$ performed best. Zaharie [3] presents a detailed exploration of the differences between *bin* and *exp* crossover regimes, the actual mutation probability in each for a given value of Cr and performance results for different settings on 50- and 100-dimension separable and non-separable versions of the frequently-used Rastrigin and Griewank problems. Cr is varied in ten steps across its range. For problems other than the rotated Griewank instance, performance at low and high values of Cr was good (with low Cr performing better on the separable problems, less well on non-separable problems) and poor around $Cr \in [0.5, 0.7]$.

The next sections examine properties of moves generated when the number of components mutated in a *target* vector is varied, followed by an examination of DE's performance as Cr is varied on a suite of problems with different search landscapes. Deductions from the move analysis and performance evaluation are given in Section III-A before the role of the other parameters F and Np is examined.

II. NUMBER OF DIMENSIONS MUTATED AND MOVE CHARACTERISTICS

In the current analysis and subsequent examination of DE's performance with varying Cr , six 50D benchmark functions were used: Hyper-ellipsoid (f_1), Schwefel (f_2), Rotated hyper-ellipsoid (f_3), Rosenbrock (f_4), Fletcher-Powell (f_5) and the FastFractal "DoubleDip" function (f_6) from the CEC2008 LSGO competition [6].¹ f_1 and f_2 are separable, f_2 , f_5 and f_6 are multi-modal, and f_3 is a non-separable but unimodal rotated version of f_1 . f_5 and f_6 have irregular search landscapes, especially f_6 , which is fractal.

The Cr parameter of DE controls the number of dimensions in a newly generated solution that are mutated with respect to the *target* solution. The number of mutated dimensions is denoted by m hereafter. Considering the *bin*

variant, values other than the extremes of 0 and 1 provide access to range of values of m surrounding approximately $Cr \cdot D$, where D is the number of dimensions.² At the extreme values either $m = 1$ (specified by the randomly selected I_i in Equation 2 above) when $Cr = 0$ or $m = D$ when $Cr = 1$. In the following analysis of exploratory moves m is used as an analogue of Cr .

In order to examine characteristics of moves that may be generated by a DE search a simulator was implemented to sample moves similar to those normal DE can generate. Each sample is derived in the following manner. First, a random *target* solution is created. A randomised vector v is then generated and m components are added to the *target* to produce a new solution, u . The magnitude of v is selected with uniform probability from $[mag_l, mag_u]$, which are expressed as fractions of the main space diagonal of the solution space, denoted by d . In terms of the DE equations given above, this is equivalent to using the *target* as the *base* and replacing the index I_i that must be mutated with a set of m indices (and setting $Cr = 0$ so that only those indices are mutated). At least during the initial iterations of a DE algorithm, and often for the entire search, the direction of difference vectors is random, as in the simulator. The following properties of the move from the *target* to u are recorded:

- its magnitude (as a fraction of d);
- the value of $f(\text{target})$ and $f(u)$;
- the change in value $\Delta f(x) = f(u) - f(\text{target})$; and
- whether or not it would be accepted, i.e., if $\Delta f(x) \leq 0$.

The new solution u is discarded before a new sample is produced; it does not become a future *target*. The other key distinction between the moves generated by the simulator and those generated during a DE search is that the simulated *target* solutions are selected randomly (in DE their quality will improve over time) yet the magnitude of the vector added may be small or large (in DE small difference vectors result when the population has contracted). Aggregate statistics were generated by allocating measurements to bins of size $0.05 \cdot d$.

Moves were generated using four combinations of settings corresponding approximately to DE with $Cr = 0$, $Cr = 0.5$, $Cr = 1$, and $Cr = 1$ with the $mag_u = 0.15 \cdot d$ equal to the maximum observed magnitude of moves when $Cr = 0$. Hereafter these groups of settings are denoted by $m = 1$, $m = \frac{D}{2}$, $m = D$ and $m = D, mag_u = 0.15$. In all cases $mag_l = 0.001 \cdot d$, while in the first three cases $mag_u = 0.6 \cdot d$, which corresponds with the typical maximum distance between solutions in normal DE in a 50D space. For each combination of problem and simulator settings 500,000 samples were generated.

It is worth noting that $m = D, mag_u = 0.15$ does not correspond to any current DE variant, as they all apply the difference vector to a *base* that is not the *target*. Reducing the scale factor F does not result in small moves from the

¹The shorthand names $f_1 - f_6$ are used to make figures more compact and do not necessarily correspond to those functions' designations in other works. FastFractal (f_6) was designated f_7 in the CEC2008 LSGO competition [6].

²See Zaharie [3] for accurate relationships between Cr and the probability of different numbers of components being mutated.

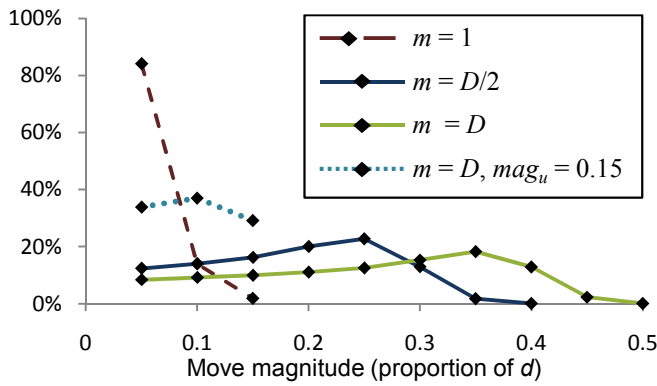


Fig. 1. Approximate distributions of move magnitudes produced by different combinations of m and mag_u

target's location as the *base* may be located far from the *target*, so a large move results [7], [8]; using too small a value of F thus leads to premature convergence [9].

Fig. 1 presents approximations of the distributions of move magnitudes for each combination of settings; the observed proportions are given at the lozenges, connecting lines are provided as visual aids. These distributions are identical for each problem. It is clear that the magnitude of moves when $m \geq \frac{D}{2}$ is generally considerably higher than when $m = 1$. When $m = 1$ the bulk of moves generated are very small.

Due to space restrictions Figs. 2 and 3 show results for a subset of the problems studied: the separable and uni-modal f_1 , rotated uni-modal f_3 and multi-modal and non-separable f_6 . The corresponding figures for f_2 , f_4 and f_5 are available online.³ In Figs. 2 and 3 the plots f_6 are representative of those for the multi-modal f_2 and f_5 , while those for f_1 are representative of those for the non-separable but uni-modal f_4 . This suggests that search landscape modality is a greater distinguishing factor than separability.

The top row of Fig. 2 shows the proportion of successful (i.e., improving) moves within each move magnitude bin, by problem. There are two features of note: the shape of the curves for each kind of problem and the degree of similarity between moves generated in different ways. With regards to the first, with uni-modal non-rotated problems the probability of a successful move decreases with move magnitude, which is to be expected since there is only a single optimum to be found and hence a global gradient heading towards it. On multi-modal problems the probability of success is similar across all move magnitudes, as there are regions both near and far that may be improving. The probability of success on the rotated f_3 shows a mixture of characteristics between the previous two. Notably, across all problems the probability of success is almost identical for moves of equivalent magnitude, regardless of the number of dimensions altered in their generation.⁴

The actual likelihood of a successful move being made

³<http://www.ict.swin.edu.au/personal/jmontgomery/research/de>

⁴The only exception is f_4 , in which $m = 1$ does show an increased probability of success of its narrow range of moves.

TABLE I
PERCENTAGE OF ACCEPTED MOVES BY PROBLEM AND MOVE
GENERATION SETTINGS

| Problem | $m = 1$ | $m = \frac{D}{2}$ | $m = D$ | $m = D, mag_u = 0.15$ |
|---------|---------|-------------------|---------|-----------------------|
| f_1 | 38 | 15 | 11 | 34 |
| f_2 | 50 | 50 | 50 | 50 |
| f_3 | 48 | 42 | 39 | 47 |
| f_4 | 39 | 12 | 8 | 28 |
| f_5 | 50 | 49 | 48 | 50 |
| f_6 | 50 | 48 | 46 | 50 |

is given by the intersection of the distributions of move magnitudes (Fig. 1) with the corresponding distributions in Fig. 2 (top). The overall probability that a move will be accepted is summarised for each combination of problem and simulator setting in Table I. Given that the bulk of moves made when $m \geq \frac{D}{2}$ have, for many of the problems, a very low probability of being accepted, the overall probability of making an improving move in such conditions is consequently small. Conversely, when $m = 1$ most solutions generated have a high probability of being accepted. Notably, the probability of accepting a move when $m = D, mag_u = 0.15$ tends to be closest to that when $m = 1$, suggesting that the direction of a move (along an axis versus at an angle to many axes) is less important than its magnitude in determining its probability of success.

The bottom row of Fig. 2 shows the average magnitude of changes in solution quality from improving moves, for each setting combination. As would be expected, larger moves can result in substantially larger improvements in solution quality. Given the way in which moves are generated, the typical improvement from $m = D, mag_u = 0.15$ is the same as that for $m = D$ over the range of move magnitudes that the first can produce. Notably, for all problems other than the rotated problem f_3 , moves generated by moving in more than one dimension have a better average improvement than those of the same magnitude in one dimension, even on the separable problems. Thus, while large moves generated with high m are less likely to be accepted, their expected improvement is very high. Better improvements from small moves in many dimensions (i.e., from $m = D, mag_u = 0.15$) are also traded off against a reduced chance of producing an improving move.

A. Probability of Further Improvement Given $f(\text{target})$

It is an implicit—but often valid—assumption in search heuristic design that good solutions are surrounded by other good solutions. This assumption is less valid or possibly invalid in fractal and fractured landscapes, in which good areas are scattered haphazardly, with little guiding global structure to exploit [10]. Nevertheless, many search heuristics explicitly or implicitly make use of gradient information; the balance between exploration and exploitation that algorithm designers seek to achieve is ultimately a balance between coarse- and fine-grained search. The nature of a search landscape will impact on the utility of different kinds of move

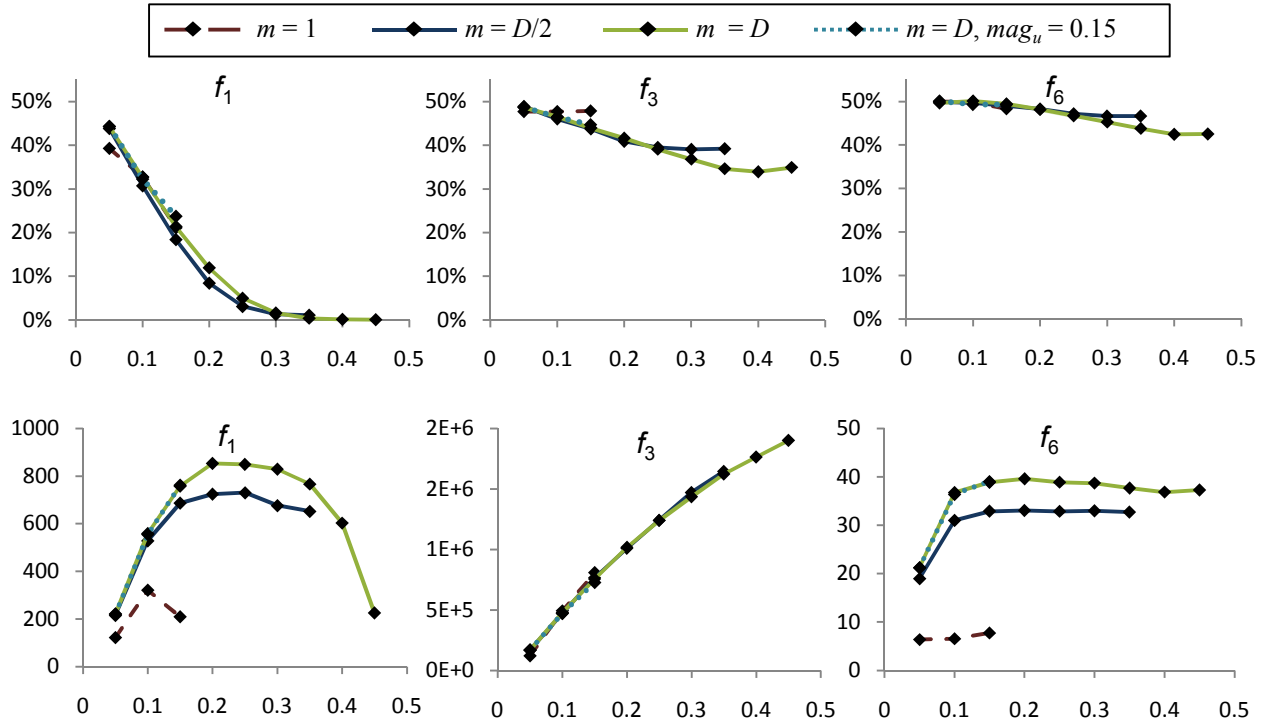


Fig. 2. Top: Probability that a move will be successful, by magnitude, for f_1 , f_3 and f_6 . Bottom: Average $|\Delta f(\text{target})|$ for improving moves, by magnitude. Horizontal axes are move magnitude as a fraction of d .

starting from solutions in particular subregions, hence having a particular initial value $f(x)$.

Fig. 3 shows the probability of making an improving move over the range of observed *target* values, for each of the simulator setting combinations. In this instance, the results for f_5 , while still very similar to those displayed for f_6 , show greater variability between the success rates of moves with $m > 1$. On f_1 (and f_4), the probability of producing an improving solution appears to be related strongly to the magnitude of moves made, with $m = 1$ and $m = D, \text{mag}_u = 0.15$ being most likely to succeed, although $m = 1$ is more likely to succeed as initial solution quality improves. On the rotated f_3 this relationship is reversed for poorer solutions, but holds for better solutions. On the multi-modal problems, the greatest probability of success from a poor starting point occurs when changing the greatest number of components, while when the initial value of a solution is below the average, changing one component at a time is most likely to succeed. The confounding effect of move magnitude (since each combination of settings has a different distribution) is partially accounted for by the results for $m = D, \text{mag}_u = 0.15$, but its probability of success from different starting points is still most similar to other approaches that change many components at a time, even though all its moves are generally small. Assuming that a small move searches within the region of a single local optima, it is possible that changing only a single dimension at a time can be more likely to succeed than a move of equivalent size in many dimensions because there are fewer directions in which it may fail [11].

B. Summary

Given that moves that change many dimensions can produce large improvements in solution quality (bottom Fig. 2) but their probability of making further successful moves from improved starting points is low (Fig. 3), especially when those moves are large (top Fig. 2), it is clearly essential that searches with high m change the magnitude of their moves once the quality of the population has been improved. In DE this means that convergence *at an appropriate rate* is required for the search to continue well. While an oft-promoted benefit of DE is that it is self-scaling [2], it appears more accurate to state that its control parameters must be set so that it scales at the correct rate; there is no implicit mechanism in the algorithm that will ensure difference vectors available at each iteration will be of the appropriate magnitude.

III. DE PERFORMANCE WITH VARYING Cr

This section presents the relative performance of DE/rand/1/bin applied to $f_1 - f_6$ as Cr is varied over the set $\{0, 0.05, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.85, 0.9, 0.95, 1\}$. Performance at each setting is measured as the average best solution produced over 25 runs. Each run is ended after $5000 \cdot D$ function evaluations. The population size (Np) and value of F are potentially confounding variables, but were set to commonly used and likely reliable values: population size is 50 (i.e., D) and $F = 0.5$, settings which have been used on problems of similar size [3] and which are unlikely to cause premature convergence [9]. Section IV considers the effects of changing both F and Np .

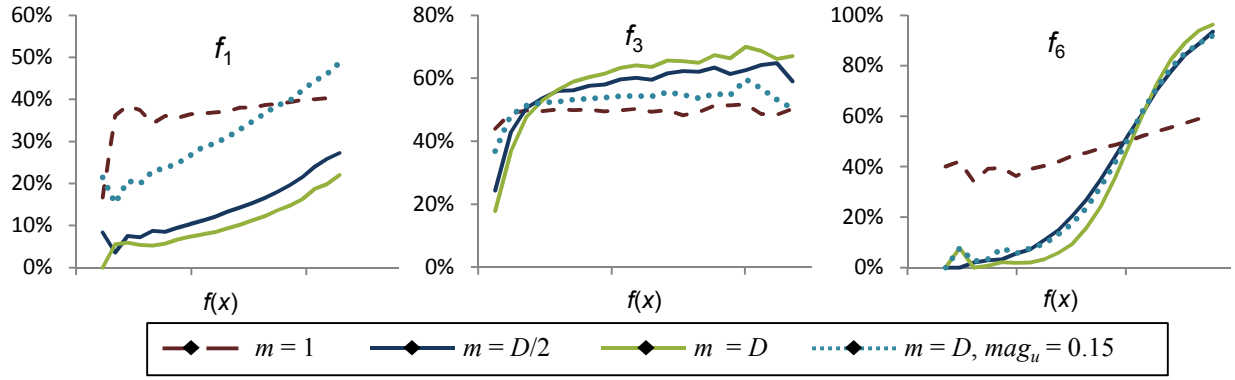


Fig. 3. Approximate probability of a move being accepted given the quality of its starting point, for f_1 , f_3 and f_6 . The $f(x)$ axes extend over the range of observed values of $f(x)$ rather than $0 - \max(f(x))$; values are not shown for compactness

Fig. 4 (top) presents the average relative performance of each Cr setting within each problem instance. Performance measures have been scaled such that 1 represents the maximum value (i.e., worst result) within Cr from $[0, 0.9]$; many results for $Cr = 1$ are too high to be meaningfully displayed. The poor performance when $Cr = 1$ is discussed in Section III-A and also in a companion work [12]. The bottom of Fig. 4 shows the overall acceptance rate of new solutions for the same runs.

The performance results are in agreement with previous [3] and current [12] work. Good performance is frequently achieved with low Cr , even on the multi-modal and non-separable problems. Cr values above 0.8 and less than 1 also produce successful searches, even though *the moves generated are very different to those when Cr is low*. Poor performance occurs for middling values of Cr . A similar pattern is observed for f_4 , although DE's performance across Cr values is generally poor. The overall success rate is generally related to the quality of the outcome, except for $Cr = 1$, where the consistently high acceptance rate appears to indicate premature convergence as individuals are attracted to one good location.

A. Move Magnitude and Convergence Rate

This section illustrates features of sample individual DE runs that are not otherwise evident from the results presented in Fig. 4. These features are considered in light of the examination of the characteristics of different kinds of move presented in Section II. Fig. 5 shows the average magnitude of moves attempted by the algorithm (left) and the average quality of population members (right) for f_2 Schwefel (top) and f_3 Rotated hyper-ellipsoid (bottom). The figure presents these measures for runs with $Cr \in \{0, 0.5, 0.9, 1\}$, which, with the exception of $Cr = 0.9$, correspond to the simulator settings $m = 1$, $m = \frac{D}{2}$ and $m = D$. The example run for f_2 is illustrative of DE applied to multi-modal landscapes, while that for f_3 illustrates differences between low and high Cr on a rotated landscape to which low Cr is ill-suited.

Regardless of the value of Cr , the magnitude of moves attempted is strongly correlated ($r > 0.99$) with the population's spread, which is to be expected since moves are

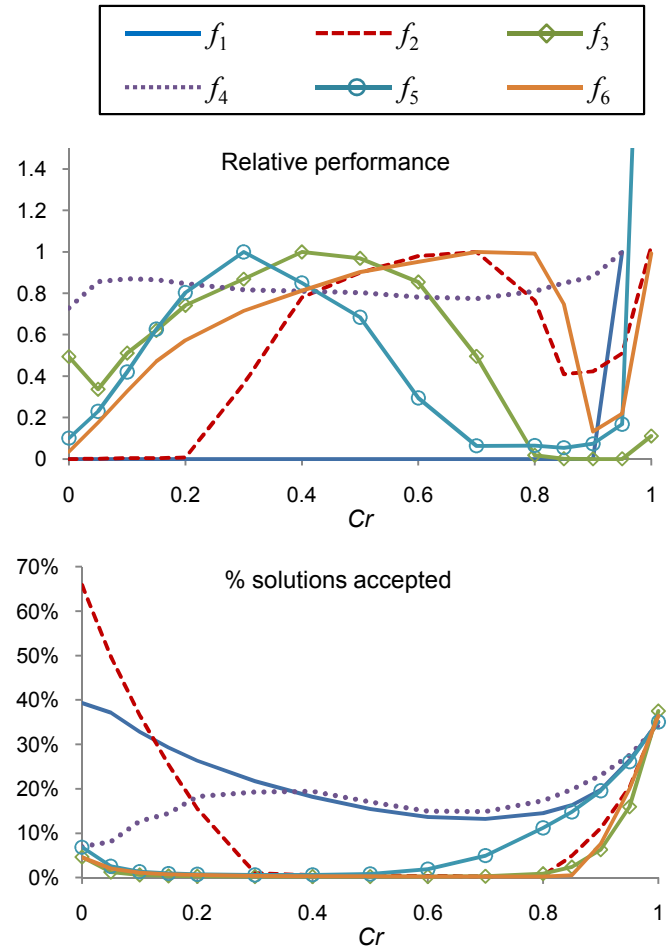


Fig. 4. Relative performance of DE/rand/1/bin (top) and total proportion of successful moves (bottom) within problem instance as Cr is varied. Performance results have been scaled such that the worst result is 1; values above 4 are not shown.

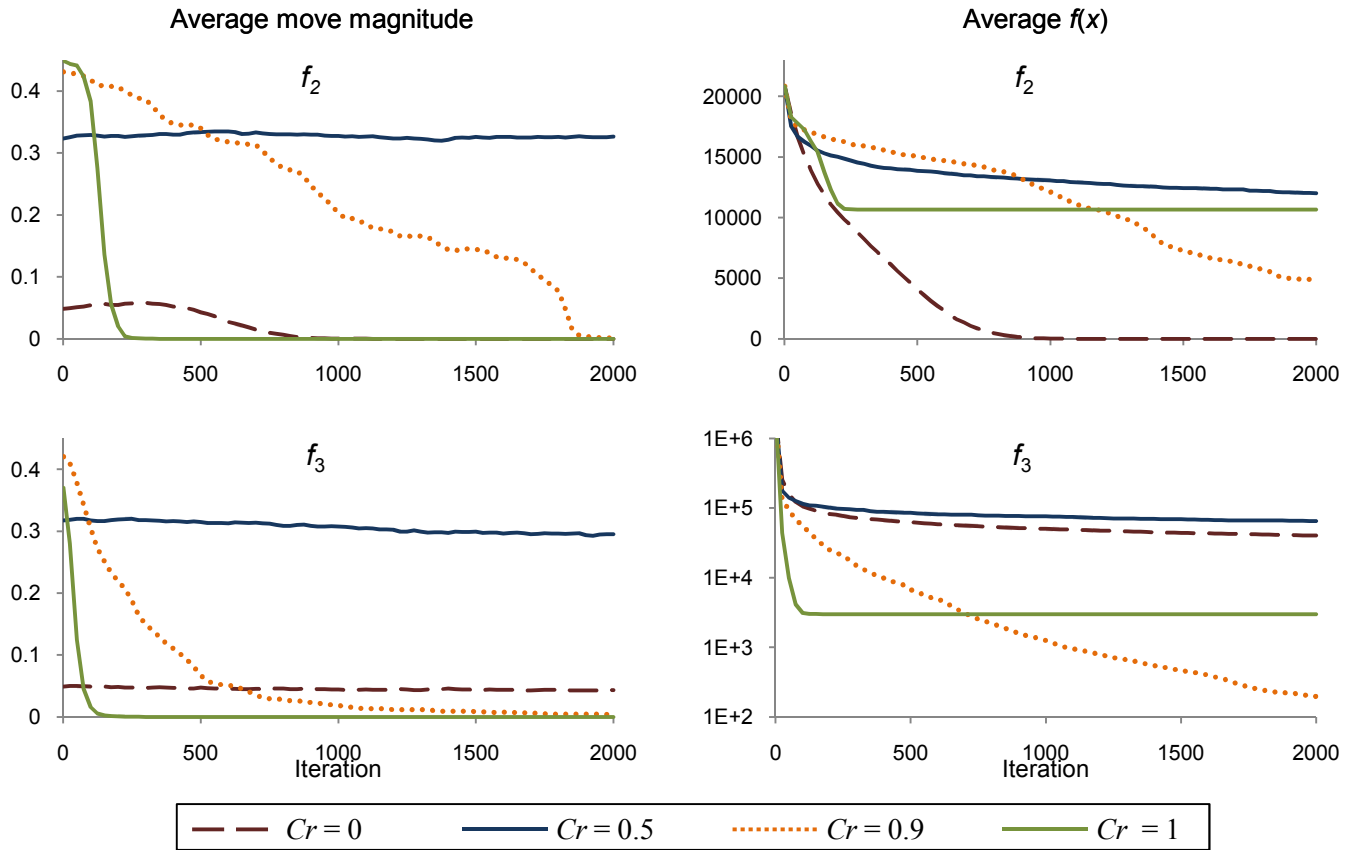


Fig. 5. Observed average move magnitude (left, as a proportion of d) and average $f(x)$ (right) for individual runs of DE on f_2 Schwefel (top) and f_3 Rotated Hyper-ellipsoid (bottom). The plotted average move magnitude has been smoothed by taking a running average over 50 iterations.

generated using difference vectors between current population members. However, although the populations are equally spread at the beginning of each run, it is clear that $Cr = 0$ makes very small moves throughout (unless it has converged). In both runs, the convergence rate (indicated by the change in magnitude of moves generated) is inversely related to the value of Cr , with $Cr = 1$ showing exceptionally fast convergence. Changes in the magnitude of moves made when $Cr \geq 0.9$ demonstrate the algorithm's ability to the automatically scale its moves.

With regards to f_2 , $Cr = 0$ is clearly best-suited for exploring its solution space, showing strong improvements in solution quality. $Cr = 0.5$ and $Cr = 0.9$ show ongoing, but less rapid improvement, while $Cr = 1$ shows rapid improvement and premature convergence. On f_3 , after initial improvements the search with $Cr = 0.5$ appears to stagnate, suggesting that the population has not converged sufficiently for subsequent moves to be scaled appropriately. $Cr = 0.9$ produces ongoing improvements in solution quality and a commensurate decrease in the size of moves made. $Cr = 0$ proceeds very slowly, despite producing the same number of improving solutions as when $Cr = 0.9$ (see Fig. 4). $Cr = 1$ again converges prematurely.

The cause of $Cr = 1$'s premature convergence lies in the nature of DE/rand/1's mutation mechanism, which generates

a new point by displacing a *base* solution that is not the *target* for replacement. Previous work [7], [8] has found that relatively small difference vectors applied to *base* solutions at some distance from their respective *targets* can contribute to population convergence, a potential problem in search spaces with many competing optima that may then not be thoroughly explored. In the case of $Cr = 1$, newly generated points are guaranteed to be within $F \cdot \|x_{r2} - x_{r3}\|$ units of the *base*. Because not all solutions (hence, not all *bases*) are replaced at each iteration, if the new solution replaces the *target* then in subsequent iterations the population variance will be diminished. Any value of Cr less than 1 thus ensures greater diversity by producing new solutions that, in some dimensions, are not located near the *base*.

In summary, when $Cr \approx 0$ DE makes very small exploratory moves, aligned with a small number of axes. The search proceeds in a gradual but consistent fashion as the likelihood of making an improving move is higher when moves are small. However, the amount of improvement with each accepted move may not be great. When $Cr \approx 0.9$ DE makes large exploratory moves that, while being less likely to be improving, can yield large improvements in solution quality *and* a reduction in the population's spread. This latter feature is clearly required so that subsequent moves are scaled appropriately for performing a more fine-grained

TABLE II
PERFORMANCE VARIATION WITH F

| f | Cr | 0.1 | 0.3 | 0.5 | 0.8 | 1.0 | Range |
|-------|-----|------|------|------|------|------|-------|
| f_1 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0.5 | 0.03 | 0 | 0 | 0 | 0.75 | 0.75 |
| | 0.9 | 1 | 0.07 | 0 | 0 | 0.2 | 1 |
| f_2 | 0.1 | 0.01 | 0 | 0.01 | 0 | 0 | 0.01 |
| | 0.5 | 0.09 | 0.07 | 1 | 0.93 | 0.57 | 0.93 |
| | 0.9 | 0.63 | 0.44 | 0.47 | 0.64 | 0.49 | 0.2 |
| f_3 | 0.1 | 0.16 | 0.19 | 0.28 | 0.37 | 0.48 | 0.32 |
| | 0.5 | 0.04 | 0.2 | 0.52 | 0.82 | 1 | 0.96 |
| | 0.9 | 0.22 | 0.03 | 0 | 0.1 | 0.65 | 0.65 |
| f_4 | 0.1 | 0.02 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| | 0.5 | 0.04 | 0.01 | 0.01 | 0.12 | 1 | 0.99 |
| | 0.9 | 0.23 | 0.03 | 0.01 | 0.01 | 0.1 | 0.22 |
| f_5 | 0.1 | 0.12 | 0.16 | 0.2 | 0.23 | 0.22 | 0.12 |
| | 0.5 | 0.09 | 0.03 | 0.35 | 1 | 0.81 | 0.97 |
| | 0.9 | 0.84 | 0.15 | 0.04 | 0.06 | 0.31 | 0.8 |
| f_6 | 0.1 | 0.15 | 0.35 | 0.38 | 0.39 | 0.36 | 0.24 |
| | 0.5 | 0.04 | 0.89 | 0.99 | 1 | 0.87 | 0.96 |
| | 0.9 | 0.46 | 0.22 | 0.17 | 0.69 | 0.35 | 0.52 |

search of the solution space. When $Cr \approx 0.5$ DE behaves similarly to $Cr \approx 0.9$, but appears unable to self-scale later moves as initial improvements do not cause the population to converge sufficiently. Thus, the exploratory moves made are neither gradual enough (as with $Cr \approx 0$) nor large enough (as with $Cr \approx 0.9$) to continue the search productively.

There are two complicating factors that must be considered. First, there clearly exist separable solution spaces that are best searched one dimension at a time; in these $Cr = 0$ will perform best because its moves will be directed either directly towards or directly away the global optimum. There are also problems, such as f_3 , where the DE population can become aligned with a key axis of the search landscape; in such situations the difference vectors will also be aligned with the landscape and high Cr will perform best. Second, the population size and F have an important role in controlling the size of moves attempted and convergence rate of the algorithm. These are discussed next.

IV. ON THE ROLE OF F AND POPULATION SIZE

Previous findings [7], [8] indicate that population convergence in DE is in large part due to its solution generation mechanism as, when Cr is high, newly generated solutions are typically nearer to the *base* than the *target*. Because F affects the distance from the *base* that a new solution is generated, it will clearly influence population convergence. As F becomes smaller and displacement from *base* solutions diminishes, good quality *base* solutions are likely to attract many other solutions to their neighbourhood [7]. When Cr is low, the number of components taken from the newly generated point is also low, so it would be expected that F has less effect. This accords with previous findings [9], in which a relation between population variance, probability of mutation (i.e., Cr) and F was derived, showing that the range of effective values of F is greater when Cr is low.

Tables II and III show the relative performance and success rate (in percent) for DE/rand/1/bin with a population

TABLE III
MOVE SUCCESS RATE VARIATION WITH F

| f | Cr | 0.1 | 0.3 | 0.5 | 0.8 | 1.0 | Range |
|-------|-----|-----|-----|-----|-----|-----|-------|
| f_1 | 0.1 | 81 | 43 | 33 | 18 | 12 | 69 |
| | 0.5 | 89 | 41 | 15 | 2 | 1 | 88 |
| | 0.9 | 93 | 46 | 20 | 2 | 1 | 92 |
| f_2 | 0.1 | 76 | 61 | 37 | 17 | 12 | 64 |
| | 0.5 | 87 | 12 | 0 | 0 | 1 | 87 |
| | 0.9 | 92 | 40 | 11 | 1 | 4 | 91 |
| f_3 | 0.1 | 1 | 1 | 1 | 1 | 1 | 0 |
| | 0.5 | 11 | 0 | 0 | 0 | 0 | 11 |
| | 0.9 | 91 | 40 | 6 | 0 | 0 | 91 |
| f_4 | 0.1 | 22 | 16 | 12 | 5 | 3 | 20 |
| | 0.5 | 65 | 26 | 17 | 1 | 0 | 65 |
| | 0.9 | 93 | 46 | 23 | 2 | 1 | 92 |
| f_5 | 0.1 | 1 | 1 | 1 | 1 | 2 | 0 |
| | 0.5 | 18 | 28 | 1 | 0 | 1 | 28 |
| | 0.9 | 92 | 46 | 20 | 1 | 1 | 91 |
| f_6 | 0.1 | 1 | 1 | 1 | 1 | 1 | 0 |
| | 0.5 | 27 | 0 | 0 | 0 | 0 | 27 |
| | 0.9 | 91 | 40 | 9 | 0 | 2 | 91 |

of 50 and $Cr \in \{0.1, 0.5, 0.9\}$, as F is varied across $\{0.1, 0.3, 0.5, 0.8, 1.0\}$. The last column shows the range of the values in each row. These results confirm that the performance of low Cr is less sensitive to the value of F than when higher values of Cr is used. However, other than with the rotated f_3 and complex f_5 and f_6 , the acceptance rate for new solutions is inversely related to F , regardless of the value of Cr used.

Population size is often cited as a source of increased diversity [5], [13], [14], but would be expected to have different impacts on searches with low versus high Cr given the differing ways in which they explore solution space. Tables IV and V show the relative performance and success rate (in percent) for DE/rand/1/bin with $F = 0.5$ and $Cr \in \{0.1, 0.5, 0.9\}$, as population size is varied across $\{50, 100, 150, 250, 500\}$. It has been recommended previously that a population size of $10 \cdot D$ be used [2], which would be 500 for these problems. There are two key features to note in these results. First, the performance at all three Cr settings tends to improve as population size *decreases*. Second, the acceptance rate of new solutions is always inversely related when $Cr = 0.9$, yet on several problems is positively related when $Cr \leq 0.5$. This suggests that a larger population can promote improved exploration when Cr is low; the use of a fixed number of function evaluations, however, means that the resulting gradual search has fewer opportunities to improve each member of the population. When Cr is high, a large population works against convergence and prolongs the time during which large difference vectors are produced. This in turn results in a greater number of inappropriately large moves being used to try to improve solutions. Consequently, reducing the value of F allows large populations to converge.

DE's different search behaviours given the values of Cr , F and population size have implications for adaptive and self-adaptive DE algorithms, which adjust these values during a run. These are discussed in a companion work [12].

TABLE IV
PERFORMANCE VARIATION WITH POPULATION SIZE

| f | Cr | Population size | | | | | | Range |
|-------|------|-----------------|------|------|------|------|------|-------|
| | | 50 | 100 | 150 | 250 | 500 | | |
| f_1 | 0.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0.5 | 0 | 0 | 0 | 0 | 0.04 | 0.04 | 0.04 |
| | 0.9 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| f_2 | 0.1 | 0 | 0 | 0 | 0.33 | 0.53 | 0.53 | 0.53 |
| | 0.5 | 0.73 | 0.79 | 0.82 | 0.84 | 0.88 | 0.15 | 0.15 |
| | 0.9 | 0.34 | 0.83 | 0.96 | 0.96 | 1 | 0.66 | 0.66 |
| f_3 | 0.1 | 0.42 | 0.47 | 0.54 | 0.57 | 0.64 | 0.22 | 0.22 |
| | 0.5 | 0.79 | 0.87 | 0.98 | 0.97 | 1 | 0.21 | 0.21 |
| | 0.9 | 0 | 0 | 0.05 | 0.27 | 0.54 | 0.54 | 0.54 |
| f_4 | 0.1 | 0.13 | 0.13 | 0.13 | 0.13 | 0.32 | 0.2 | 0.2 |
| | 0.5 | 0.12 | 0.12 | 0.12 | 0.14 | 0.62 | 0.5 | 0.5 |
| | 0.9 | 0.13 | 0.1 | 0.11 | 0.13 | 1 | 0.9 | 0.9 |
| f_5 | 0.1 | 0.12 | 0.17 | 0.19 | 0.22 | 0.33 | 0.21 | 0.21 |
| | 0.5 | 0.22 | 0.47 | 0.54 | 0.64 | 0.77 | 0.55 | 0.55 |
| | 0.9 | 0.02 | 0.02 | 0.02 | 0.73 | 1 | 0.98 | 0.98 |
| f_6 | 0.1 | 0.26 | 0.33 | 0.36 | 0.42 | 0.51 | 0.25 | 0.25 |
| | 0.5 | 0.79 | 0.83 | 0.8 | 0.84 | 0.88 | 0.09 | 0.09 |
| | 0.9 | 0.07 | 0.94 | 0.98 | 0.99 | 1 | 0.93 | 0.93 |

V. CONCLUSIONS

Common explanations of DE's search behaviour as Cr is varied focus on the directionality of the search and ignore the magnitude of moves made. An analysis of moves generated by mutating differing numbers of dimensions suggests that the probability of making a successful move is more strongly related to its magnitude (in conjunction with the quality of the starting point) than to the number of dimensions in which it occurs. Moves in many dimensions can also produce greater improvements in solution quality, even when the magnitude of those moves is the same. Low and high values of Cr can both produce effective searches: low values because each population member conducts gradual, frequently successful exploration; high values because they produce rapid improvements in solution quality and contraction of the search space, which reduces the size of subsequent moves. When the population doesn't converge rapidly enough the search stagnates. $Cr = 1$ is generally ineffective because the population converges too rapidly, an artefact of DE's solution generation mechanism. This explains why a value a little less than one is so often used.

The parameter F affects most strongly searches with high Cr , where it affects the rate of convergence. Population size affects searches at both extremes of Cr , but in different ways. When the number of function evaluations is fixed, a small population will make best use of the search provided by $Cr \approx 0$ because each individual is potentially updated more times. As each improvement is likely to be small the number of times a solution is updated is important. When Cr is high, either a small population and large value of F or a large population and small value of F can perform well, as diversity is maintained longer as F and population size increase, which may delay the necessary reduction in the size of moves made.

TABLE V
MOVE SUCCESS RATE VARIATION WITH POPULATION SIZE

| f | Cr | Population size | | | | | | Range |
|-------|------|-----------------|-----|-----|-----|-----|----|-------|
| | | 50 | 100 | 150 | 250 | 500 | | |
| f_1 | 0.1 | 33 | 33 | 32 | 33 | 33 | 0 | 0 |
| | 0.5 | 15 | 14 | 14 | 14 | 14 | 1 | 1 |
| | 0.9 | 20 | 13 | 11 | 9 | 7 | 12 | 12 |
| f_2 | 0.1 | 37 | 19 | 11 | 6 | 8 | 31 | 31 |
| | 0.5 | 0 | 1 | 1 | 1 | 3 | 2 | 2 |
| | 0.9 | 11 | 1 | 1 | 1 | 2 | 11 | 11 |
| f_3 | 0.1 | 1 | 1 | 1 | 2 | 3 | 2 | 2 |
| | 0.5 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| | 0.9 | 6 | 2 | 2 | 2 | 2 | 5 | 5 |
| f_4 | 0.1 | 12 | 14 | 14 | 14 | 14 | 3 | 3 |
| | 0.5 | 17 | 9 | 8 | 8 | 7 | 10 | 10 |
| | 0.9 | 23 | 13 | 10 | 7 | 5 | 18 | 18 |
| f_5 | 0.1 | 1 | 2 | 3 | 4 | 7 | 6 | 6 |
| | 0.5 | 1 | 1 | 1 | 2 | 3 | 2 | 2 |
| | 0.9 | 20 | 8 | 4 | 2 | 2 | 18 | 18 |
| f_6 | 0.1 | 1 | 2 | 2 | 3 | 6 | 4 | 4 |
| | 0.5 | 0 | 0 | 1 | 1 | 2 | 2 | 2 |
| | 0.9 | 9 | 0 | 1 | 1 | 1 | 8 | 8 |

REFERENCES

- [1] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.
- [2] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin: Springer, 2005.
- [3] D. Zaharie, "Influence of crossover on the behavior of differential evolution algorithms," *Applied Soft Comput.*, vol. 9, no. 3, pp. 1126–1138, 2009.
- [4] J. Rönkkönen, S. Kukkonen, and K. Price, "Real-parameter optimization with differential evolution," in *IEEE CEC 2005*. Munich, Germany: IEEE, 2005, pp. 567–574.
- [5] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *GECCO 2006*, Seattle, Washington, USA, 2006, pp. 485–492.
- [6] K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y. Chen, C. Chen, and Z. Yang, "Benchmark functions for the cec 2008 special session on large scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Technical Report, 2007.
- [7] J. Montgomery, "Differential evolution: Difference vectors and movement in solution space," in *IEEE CEC 2009*. Trondheim, Norway: IEEE, 2009, pp. 2833–2840.
- [8] —, "The effect of different kinds of move in differential evolution searches," in *4th Australian Conf. on Artificial Life*, ser. LNAI, K. Korb, M. Randall, and T. Hendtlass, Eds., vol. 5865. Melbourne, Australia: Springer, 2009, pp. 272–281.
- [9] D. Zaharie, "Critical values for the control parameters of differential evolution algorithms," in *MENDEL 2002, 8th International Conference on Soft Computing*, R. Matoušek and P. Ošmera, Eds., Brno, Czech Republic, 2002, pp. 62–67.
- [10] S. Chen and V. Lupien, "Optimization in fractal and fractured landscapes using locust swarms," in *4th Australian Conf. on Artificial Life (ACAL09)*, ser. LNAI, K. Korb, M. Randall, and T. Hendtlass, Eds., vol. 5865. Melbourne, Australia: Springer, 2009, pp. 232–241.
- [11] S. Chen and Y. Noa Vargas, "Improving the performance of particle swarms through dimension reductions — A case study with locust swarms," in *IEEE CEC 2010*. Barcelona, Spain: IEEE, 2010.
- [12] J. Montgomery and S. Chen, "An analysis of the operation of differential evolution at high and low crossover rates," in *IEEE CEC 2010*. Barcelona, Spain: IEEE, 2010.
- [13] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Comput.*, vol. 9, no. 6, pp. 448–462, 2005.
- [14] J. Brest and M. Sepešy Maučec, "Population size reduction for the differential evolution algorithm," *Applied Intelligence*, vol. 29, no. 3, pp. 228–247, 2008.