Differential Evolution: Difference Vectors and Movement in Solution Space

James Montgomery

***Abstract*—In the commonly used DE/rand/1 variant of differential evolution the primary mechanism of generating new solutions is the perturbation of a randomly selected point by a difference vector. The newly selected point may, if good enough, then replace a solution from the current generation. As the magnitude of difference vectors diminishes as the population converges, the size of moves made also diminishes, an oft-touted and obvious benefit of the approach. Additionally, when the population splits into separate clusters difference vectors exist for both small and large moves. Given that a replaced solution is not the one perturbed to create the new, candidate solution, are the large difference vectors responsible for movement of population members between clusters? This paper examines the mechanisms of small and large moves, finding that small moves within one cluster result in solutions from another being replaced and so appearing to move a large distance. As clusters tighten this is the only mechanism for movement between them.**

I. Introduction

The differential evolution (DE) algorithm [1] is a robust and widely effective technique for optimisation in continuous domains. It is robust both in terms of the problems to which it has been applied and in that it shows some degree of insensitivity to the values of its governing parameters [2], [3]. Despite a plethora of studies describing its application to specific problems, there have been relatively few analyses of the mechanisms underlying its behaviour (notable examples are given in the next section) [4]. The DE heuristic employs a population of solutions, each of which is considered as a *target* for replacement by a newly generated solution to produce a subsequent generation of solutions. New candidate solutions are generated by the addition of one or more scaled difference vectors to some member of the population, followed by uniform crossover with the solution that may be replaced by the candidate.[1] Variants of the algorithm result from different ways of generating difference vectors, how many difference vectors are employed and the amount of crossover used. DE algorithms are typically named according to a system that largely reflects these characteristics.

The three main variants of DE are designated DE/rand/1/bin, DE/best/1/bin and DE/target-to-best/1/bin, with each of these employing one difference vector generated from members of the population and making a biased decision concerning crossover for each solution component (the number of which consequently follows a binomial distribution). Crossover can alternatively involve sequences of vector components, the lengths of which follow an inverse exponential distribution. As the *bin* crossover is a superset of *exp*, this is the form of crossover considered in the remainder of the paper. The DE/rand/1 variant is most common and is used to illustrate the algorithm's operators. Let $S = \{x_1, x_2, \ldots, x_{|S|}\}$ 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

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

where x_{r1}, 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. In this paper x_{r1} is also referred to as the *base*. Uniform crossover is performed on v_i, controlled by the parameter $Cr \in [0, 1]$, to produce the trial solution u_i, according to

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

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 $u_i \neq x_i$. In the "best" variant the best member of S (denoted *best* hereafter) replaces the *base* in Equation 1. In the "target-to-best" variant the *target* replaces the *base* solution, and Equation 1 incorporates an additional scaled difference vector from the *target* to the *best*. In all cases the *target* is replaced if the new solution is as good or better; in this way the algorithm continues to move in flat areas of a problem landscape. The study presented in this paper considers only the DE/rand/1/bin variant, as it is commonly used and has been found to be widely effective [5].[2]

One of the intuitive benefits of employing difference vectors is that the magnitude of moves decreases as the population converges to an optimum, allowing a more fine-grained search to occur [6]. Additionally, when multiple optima exist and the population splits into separate clusters around these, some difference vectors will be small (intra-cluster) and some will be large (extra-cluster); yet does extra-cluster movement occur because of this? This paper examines the underlying causes of both large and small movement of solutions with respect to the *target* to determine if this intuitive understanding of DE's behaviour is correct.

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

[1]There are also variants which perform *arithmetic crossover*, but these are not considered in this study.

[2]DE/best/1/bin has also been found to be widely effective [5] but is not included in detail due to space constraints.

A. Previous Analyses of DE's Emergent Behaviour

As with many complex systems, the emergent behaviour of the DE algorithm from its low-level operators cannot be reliably predicted or, rather, the reliability of such predictions is due to the algorithm's robustness. Price, Storn and Lampinen [6] provide a quite extensive analysis of the performance of different DE variants with different values for the key parameters F and Cr (and others in the pertinent variants). Results are presented as phase portraits, with successful combinations plotted on 2D maps of the space of values for F and Cr. While of undoubted benefit in demonstrating the algorithm's insensitivity (in some cases) to the values of these parameters and in providing a guide to suitable values for different kinds of problem, such portraits cannot explain *why* the algorithm is successful or not. In a similar vein, Zaharie [7] analysed the effect of the value of F on the convergence behaviour of DE, concluding that higher values work against premature convergence; what is an appropriately high value, however, is somewhat problem-dependent.

Recently, Dasgupta, Biswas, Das and Abraham [4] developed a mathematical model of DE's behaviour and used this to predict its convergence behaviour when the population is located near an optimum. Their model is highly constrained, however, restricted to a one-dimensional problem space with the population located on a gradual slope leading to the optimum. Ali [8] derives a probability density function of the location of new solutions. This is used in a simulation to show the likely distribution of generated solutions over time when DE is applied to a particular problem, chiefly to illustrate the large number of generated solutions that lie outside the problem's bounds.

II. Analysing Movement

The difficulty in understanding a realistic DE implementation by analysing its low-level operators suggests that an alternative approach must be taken. This study takes observations of the relationships between the (randomly selected) solutions involved in producing new, trial solutions and relates this to the macro behaviour of the algorithm.

To analyse the movement of solutions is it necessary to define what it is relative to. In an algorithm that maintains and modifies a single solution this is straightforward: each modified solution has moved relative to the solution from which it was produced. In a population-based algorithm this is less straightforward, especially in the case of DE/rand/1 where a new solution is produced primarily by the displacement of the *base* solution's position, *yet the base solution is not replaced by this new solution*. More generally, an algorithm's search may be considered in terms of the new solutions it produces or changes to (movement of) the population. With regards to the new, trial solutions the three DE variants described above are coarsely characterised in Table I, split by the amount of crossover employed. Low values of Cr result in searches from the *target* or *best* (in DE/best) solution along one or more axes, while higher values of Cr result in searches from whichever point is serving as the *base* at an angle to most or all axes.

TABLE I
Characterising exploration by DE variant and Cr value in terms of current population

Variant	Low Cr: parallel to axes…	High Cr: at angle to axes…
rand	from *target*	from *base*
best	from *best*	from *best*
target-to-best	from *target*	from *target* towards *best* (with some perturbation)

Thus, at the solution generation level, movement can be viewed in relation to these alternative starting points. However, while the search (as a whole) is defined by the new solutions produced, the solutions that can be generated are restricted by the population at each iteration; the population, in turn, only changes as *target* solutions are replaced. Hence, the *replacement* of target solutions is highly relevant to the evolution of the search and to large-scale population dynamics, so it is also reasonable to consider movement in terms of the location of a trial solution with respect to the *target*, which is what is done in the remainder of this paper.

A. Classifying Moves

In the DE/rand/1 algorithm, the size of a move away from a target solution is primarily a function of the scaling factor F (typically high), the magnitude and direction of the difference vector $||x_{r2} - x_{r3}||$ and the location of the *base* solution. It is also affected by the amount of crossover performed (magnitude of Cr). The potentially confounding effects of crossover are largely ignored in the remainder of this study for the following reasons. It is typically set to a high value; small values are appropriate only for separable functions [1] which are less challenging and realistic than functions with interdependent variables. Equation 1 is thus largely responsible for determining the location of trial solutions. Further, of particular interest are the causes of large moves when the population has split into isolated clusters. The lower the value of Cr—hence the fewer components of the trial solution are taken from the solution generated by Equation 1—the smaller the move will be. While the effects of crossover will not be considered further, setting Cr to a value a little less than 1 (as opposed to exactly 1) has been observed to offer some advantage to the search, so in the experiments described in Section III it is set to the commonly used value of 0.9 so that the typical evolution of the population can be observed.

In this study, vectors (consequently, moves) are classified as either *small* or *large*, with *large* moves considered to be those greater than 20% the magnitude of the space-diagonal of the solution space. A *small* move may result from a *base* solution near the *target* perturbed by a small difference vector, or a farther *base* solution perturbed by a large difference vector. Additionally, *small* moves can result from the foreshortening of a difference vector that extends

$$||x_i - u_i|| \begin{cases} \textit{small} \begin{cases} ||x_{r2} - x_{r3}|| \text{ is } \textit{small} & \rightarrow \textit{real small} \\ ||x_{r2} - x_{r3}|| \text{ is } \textit{large} & \rightarrow \textit{fake small} \end{cases} \\ \textit{large} \begin{cases} ||x_{r2} - x_{r3}|| \text{ is } \textit{large} & \rightarrow \textit{real large} \\ ||x_{r2} - x_{r3}|| \text{ is } \textit{small} & \rightarrow \textit{fake large} \end{cases} \end{cases}$$

Fig. 1. Characterising exploration by DE/rand/1 in terms of distances between solutions involved in generating trial solutions

outside the bounds of a constrained search space or the use of small values of Cr, especially in low-dimensional spaces. The first (a small move caused by a small difference vector) is what one intuitively expects of the algorithm, while the latter examples are less likely to be considered as major causes of small moves. Conversely, *large* moves may result from a large difference vector producing a point far from the *target* or a small difference vector applied to a *base* solution that is far from the *target*. Figure 1 represents this classification of moves, with moves and difference vectors of similar magnitude being labelled "real" and those with dissimilar magnitudes being labelled "fake".[3]

III. Experiment Design and Results

The study examined the relative proportions of each class of solution described above over the course of an algorithm's run. The classifications described in the previous section are each further split into *good* and *bad*, where a *good* solution would replace the *target* and a *bad* one would not. Of particular interest are the *real* and *fake large* moves in situations where the algorithm's population has split into separate clusters. The current, apparent consensus is that large difference vectors generated between members of different clusters allow movement of solutions between the clusters (a *real large* move), although it is also possible for small moves to occur within one cluster, with the new solution replacing a target in a remote cluster (a *fake large* move).

Cluster membership is decided based on the distance between pairs of solutions; if the distance between solutions x_i and x_j is less than a threshold d (the current median distance between all solutions multiplied by F) then x_i and x_j are considered to belong to the same cluster. Thus a cluster may be wider than d but any solution in it will likely be able to reach at least one other solution in the same cluster.

A standard DE/rand/1/bin algorithm was used with $Cr = 0.9$ and $F = 0.8$, values which are commonly used and frequently effective [1], [6].[4] Illustrative results for different values for Cr and F, however, are given in the following subsection for the Peaks function. In all cases a population of 50 solutions was employed. The problems to which the algorithm was applied are given in Table II together with the number of solution evaluations (consequently, number of iterations) allowed. These limits were chosen so that, in most runs, the population has converged to a single location by the end of a run. The problems are ordered in Table II based some commonality in characteristics: Beale and Peaks are non-separable and unimodal, Branin's rcos and Himmelblau have multiple global optima, Shekel's Foxholes and Schwefel are separable functions with many local optima, the remainder are non-separable functions of a variable number of dimensions.

TABLE II
Problems used, allowed function evaluations and resultant number of iterations given population size of 50

Problem	n	Optima	Evaluations	Iterations
Beale	2	1	2000	40
Peaks	2	1	2000	40
Branin's rcos	2	3	10000	200
Himmelblau	2	4	10000	200
Shekel's Foxholes	2	1	7500	150
Schwefel	10	1	60000	1200
Langerman	2, 10	1	6000, 80000	120, 1600
Modified Shekel's	2, 10	1	4000, 25000	80, 500
Shubert	2, 10	many	35000, 350000	700, 7000

The data collected at each iteration are the number of generated solutions in each of the eight categories and an estimate of the number of clusters in the population. The algorithm was applied to each problem 100 times. As the exact proportions of different solutions can vary—the general shape remains the same—the data were smoothed by taking the mean values at each iteration. As the data are presented graphically, they were further simplified by aggregating these mean values over iterations from each fifth of the algorithm's run.

A. Sensitivity Analysis

As the relative number of generated solutions in each class will vary with the threshold chosen for discriminating between *large* and *small* moves and with the differing behaviour of the algorithm induced by different values of Cr and F, a sensitivity analysis was performed with the Peaks function. This two-dimensional (minimisation) function has one global optimum with a nearby local optimum and raised peak, with smooth transitions between them. The threshold for *small* and *large* jumps was held constant, while Cr was varied between low (0.1) and high (0.9) values and F was varied across the set $\{0.2, 0.5, 0.8\}$. The number of function evaluations was expanded to 2500 as runs with low Cr take longer to converge. Figure 2 shows the distributions of different kinds of move in blocks of 10 iterations, as well as the mean number of clusters at each iteration (overlaid white line). Note that the axes for the mean number of clusters are not of the same scale in each chart.

For $F = 0.2$ (top two charts in Figure 2) there are no *fake small* or *real large* moves, as the value of F restricts all difference vectors to be below the threshold for a large move. *Fake large* moves do occur, however, constituting slightly less than half of all moves and approximately one third of successful moves early in both runs. In this problem the population very quickly finds the basins of both the optimum and local optimum (in this period the number of local clusters

[3]Negative connotations of the word "fake" are *not* implied.

[4]An initial sensitivity analysis also found these to be effective on all problems used in this study.

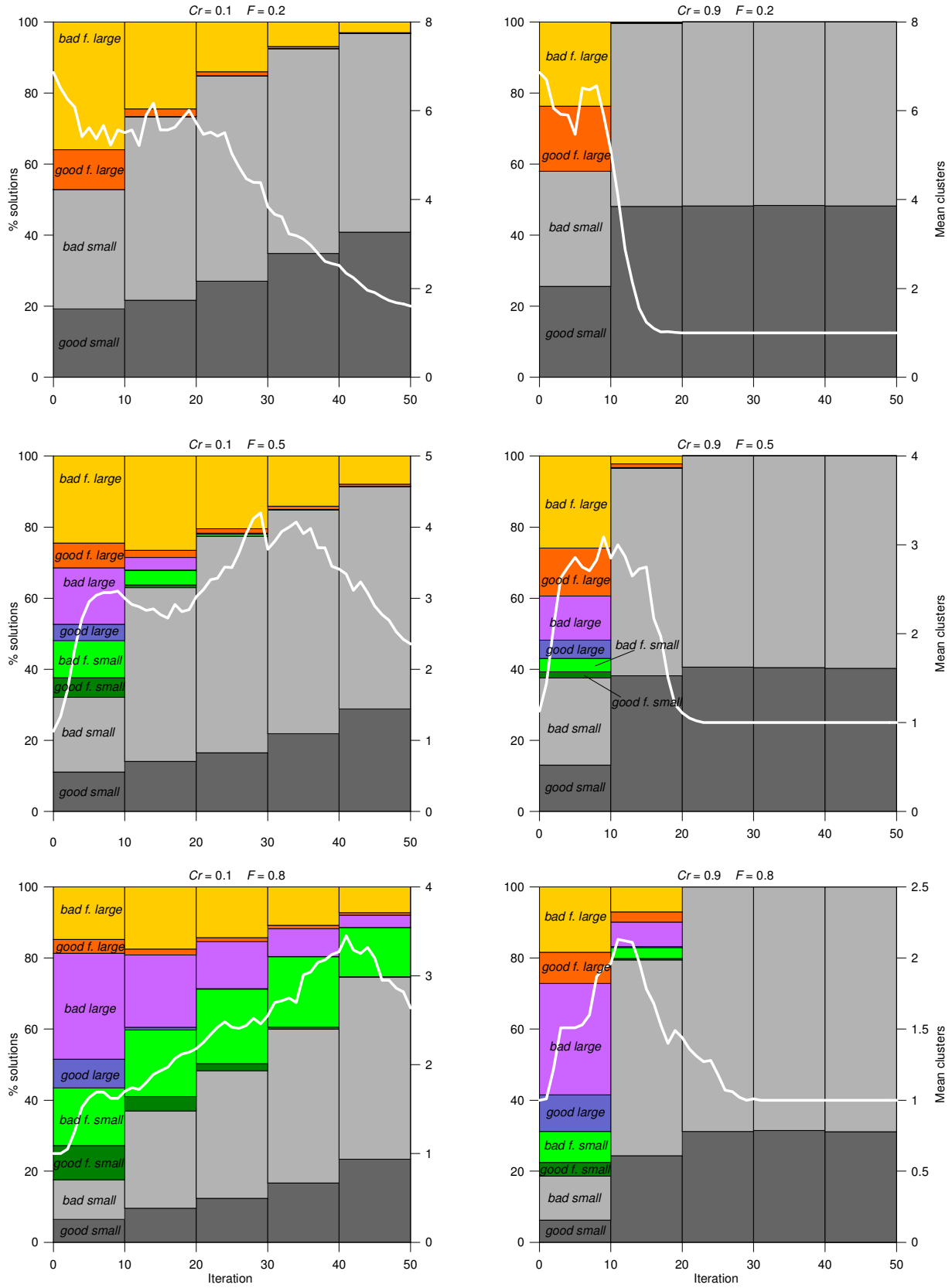

Fig. 2. Move distributions and mean number of clusters for DE/rand/1/bin applied to Peaks function, $Cr \in \{0.1, 0.9\}, F \in \{0.2, 0.5, 0.8\}$. Classes in each figure are, from bottom, *good small*, *bad small*, *good fake small*, *bad fake small*, *good large*, *bad large*, *good fake large*, *bad fake large*; Good and bad *fake small* and *real large* are absent for $F = 0.2$

is high). With low crossover movement between the local and global optimum is more difficult as they are not axis-aligned; with high crossover typically all solutions have been pulled into the global optimum's basin within the first 10 iterations.

A similar pattern of *fake large* moves is seen with $F \in \{0.5, 0.8\}$: low crossover slows convergence to the global optimum and so (*fake* and *real*) *large* moves continue to occur, while high crossover leads to fast convergence to the global optimum. As would be expected, the higher the value of F the greater the number of *large* moves made. In the case of $Cr = 0.9$, $F \in \{0.5, 0.8\}$, *fake large* moves constitute a high proportion of successful moves made early on (40% and 30% respectively in the first fifth of iterations). For the same settings, *fake small* moves constitute a much smaller proportion of all moves made.

B. Move Charts by Problem

This section presents the results for the typical DE/rand/1/bin algorithm with $Cr = 0.9$ and $F = 0.8$. Figure 3 presents the results for five 2D problems and one separable 10D problem (Schwefel). Figure 4 presents the results for three problems with a variable number of dimensions: on the left are results for their two-dimensional versions and in the right column are results for their 10-dimensional versions.

1) 2D Problems and 10D Schwefel: Results for the Beale function, which features a fairly flat search landscape for much of the search area, are similar to those for Peaks: there is an initial spread of move sizes and of both *real* and *fake* moves, *fake* moves initially account for a substantial portion of successful moves. As the number of large moves diminishes, the contribution of *fake large* and *fake small* moves drops sharply.

Branin's rcos and Himmelblau both feature several global optima, around which the population clusters in ever-tightening groups, exhibited by the rapid rise in the number of detected clusters. The proportion of successful *real large* moves quickly drops to zero, yet the number of clusters falls, indicating that movement between the clusters does occur. Throughout the searches in both problems, the proportion of *fake large* moves does not drop to zero; allowing the algorithm to run longer eventually results in convergence to just one of the global optima, at which point such moves no longer occur. Clearly, however, as the population splits into clusters around each optimum the mechanism for movement between these is *not* the large difference vectors generated by pairs of solutions in competing clusters. The reason for this is that, for $F < 1$, as tight clusters form moves between them fall short of the region surrounding each cluster where the new solution would be accepted, not venturing far enough into their respective basins of attraction to be kept.

Shekel's Foxholes and Schwefel both exhibit a low proportion of successful moves (of any kind). In addition to some successful *real large* moves, Shekel's exhibits a similar number of successful *fake large* moves; as the search progresses these represent the only successful large moves made. In Schwefel, *real large* moves constitute most successful moves; as the population splits into clusters around the local optima the only successful large moves are *fake*. The chaotic mean cluster line is due to variability in exploration behaviour between runs: the peak number of clusters within a run was generally greater than 20, with a mean of four over the entire course of a run.

2) Move Classes with Increasing Dimensions: The Langerman function (the one used in this study the superposition of seven Gaussian distributions) presents a complex search landscape. In both the two- and 10-dimensional cases, a high number of *large* moves ultimately gives way to only *small* moves as the search contracts to one of the competing (local) optima. In the 2D version, in early iterations, *fake large* moves account for a substantial portion of all successful moves, while later they account for some successful moves while *fake small* and *real large* produce no successful moves (*real small* moves account for all other improvement). In the 10D version the beneficial impact of *fake large* moves is diminished, although they are still responsible for some of the unsuccessful trial solutions produced. The chaotic tail of the mean number of clusters is due to the brief (less than 10 iterations) formation of a high number of small, tight clusters in local optima very near the optimum, occurring at varying times between runs.

The Modified Shekel's function is a more chaotic variant of Shekel's Foxholes. In the 2D variant, early in the search, *fake large* moves produce more good moves than *real large* and *fake small*, and as many good moves as *real small*. By the second fifth of the run some of the good moves are *fake large*, with the rest being *real small*: *real large* and *fake small* produce no good solutions. In the 10D variant the impact of *fake large* moves is substantially diminished.

The Shubert function has a large number of competing global optima (18 in the 2D variant, increasing with the number of dimensions). In the 2D version, as with the Branin's rcos and Himmelblau problems, *fake large* moves account for a large number of successful moves as the population clusters around typically three of the alternative optima. Notably, 15–20% of moves are *fake small*, probably a result of the arrangement of optima in closely located pairs: large moves may place a new solution near to the *target* around some other optima. In the 10D version the majority of moves are *real large* (*fake large* and *fake small* account for less than 10% of moves), although very few moves of any kind are succesful. The peak number of clusters in both problems is higher than the plots suggest, as these occur at different times in different runs. In the 10D version, due to even greater variability between runs, although the number of detected clusters typically peaked between four and 10 the average number of clusters presented in the figure is chaotic and less than two.

C. Contributions to Solution Quality by Move Class

While the mechanisms of *real* and *fake large* moves differ, the improvement in solution quality produced is comparable, when considered across problems. *Fake small* moves also

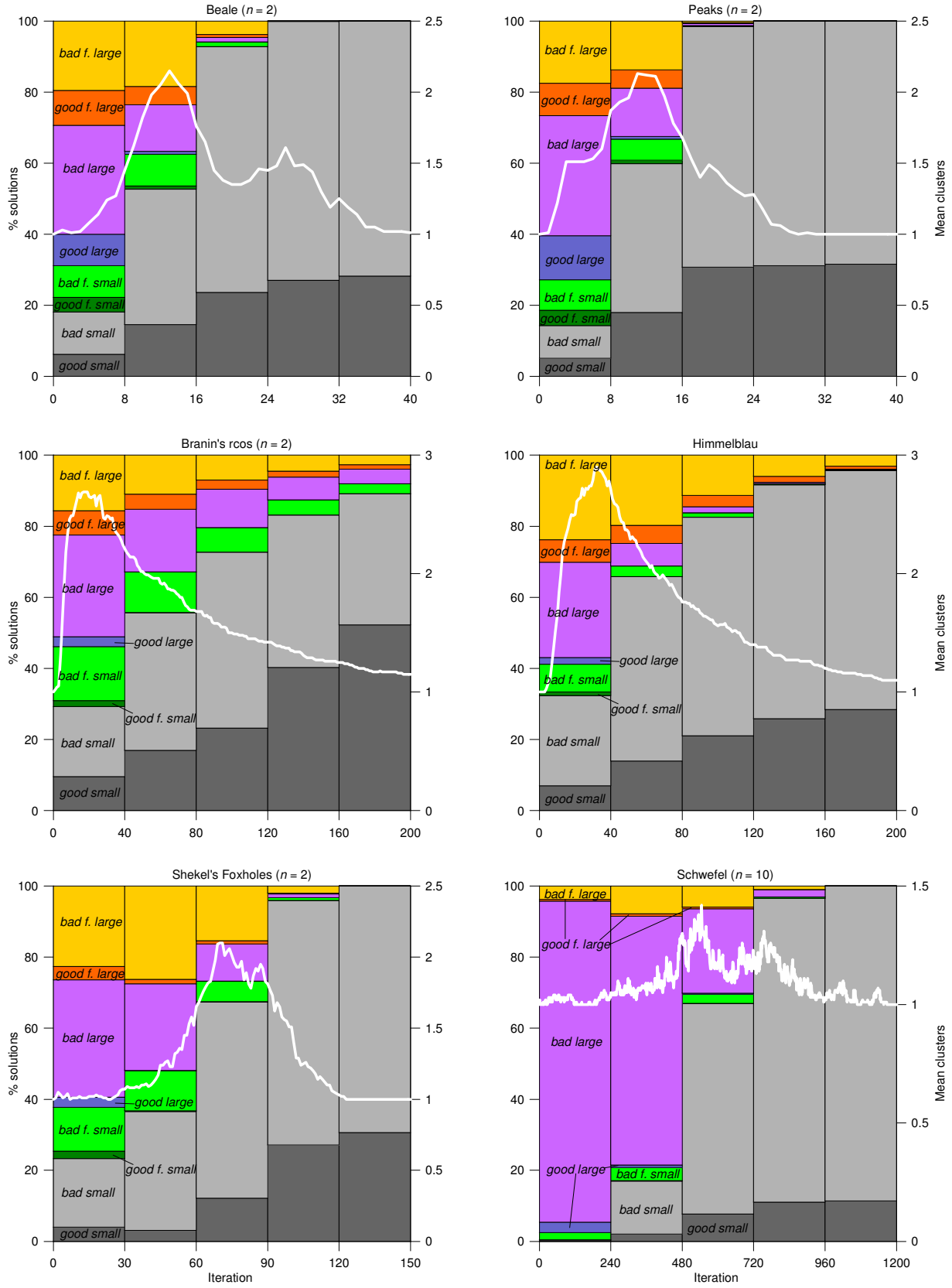

Fig. 3. Move distributions and mean number of clusters for Beale, Peaks, Branin's rcos, Himmelblau, Shekel's Foxholes and Schwefel (10D). Classes in each figure are, from bottom, *good small*, *bad small*, *good fake small*, *bad fake small*, *good large*, *bad large*, *good fake large*, *bad fake large*

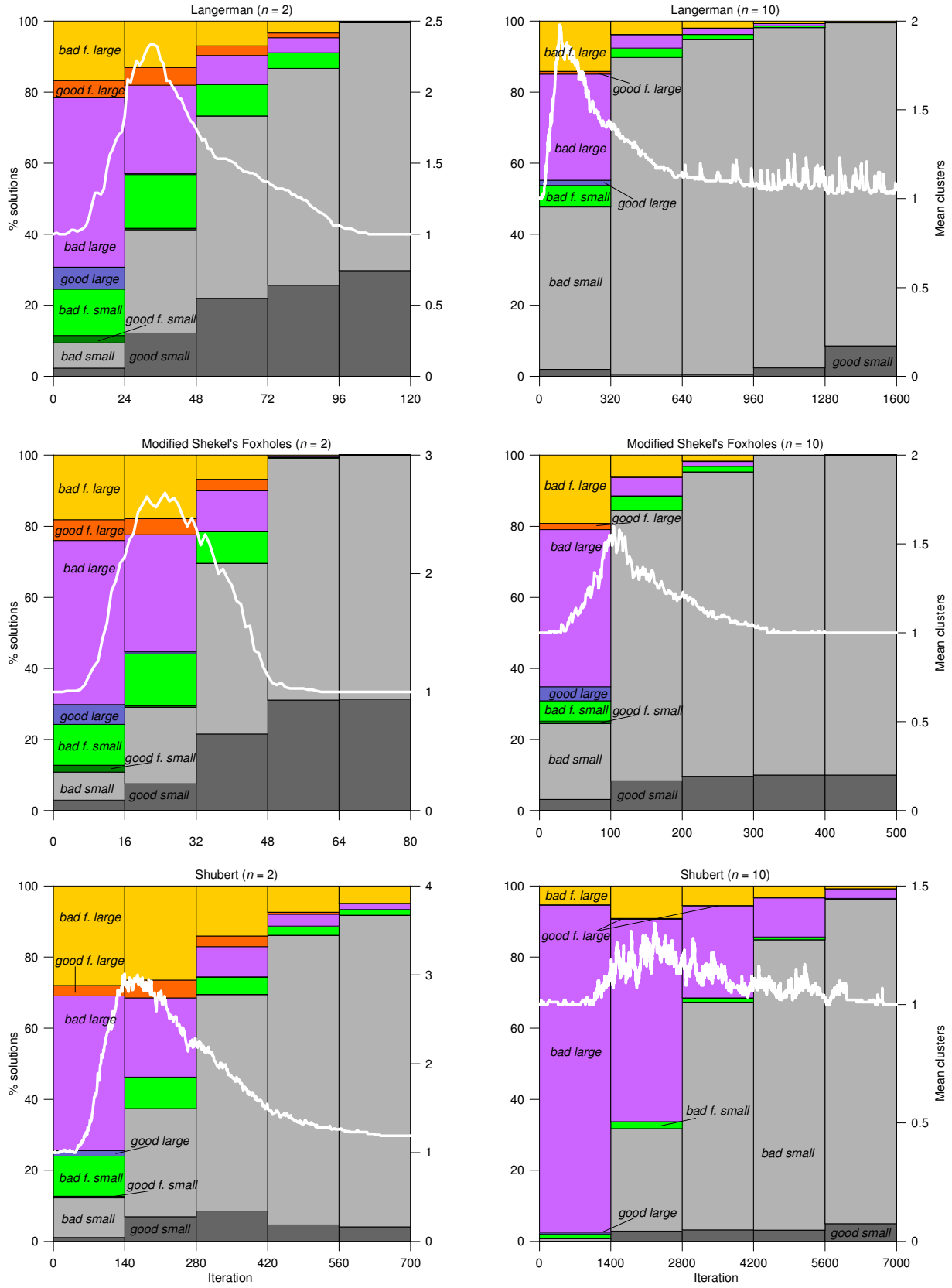

Fig. 4. Move distributions and mean number of clusters for Langerman (2D and 10D), Modified Shekel's (2D and 10D) and Shubert (2D) and 10D). Classes in each figure are, from bottom, *good small*, *bad small*, *good fake small*, *bad fake small*, *good large*, *bad large*, *good fake large*, *bad fake large*

TABLE III
ORDER OF IMPROVEMENT MAGNITUDES BY MOVE CLASS

Problem	Improvement magnitude order
Beale	*large* ≻ *f. small* ≻ *f. large*
Peaks	*f. large* ≻ *f. small* ≻ *large*
Branin's rcos	*large* ≻ *f. small* ≻ *f. large*
Himmelblau	*large* ≻ *f. small* ≻ *f. large*
Skekel's Foxholes	*f. small* ≻ *f. large* ≻ *large*
Schwefel (10)	*f. small* ≻ *f. large* ≻ *large*
Langerman (2)	*f. large* ≻ *f. small* ≻ *large*
Langerman (10)	*large* ≻ *f. large* ≻ *f. small*
Modified Shekel's (2)	*f. large* ≻ *f. small* ≻ *large*
Modified Shekel's (10)	*f. small* ≻ *large* ≻ *f. large*
Shubert (2)	*f. large* ≻ *f. small* ≻ *large*
Shubert (10)	*large* ≻ *f. small* ≻ *f. large*

contribute greatly to improvements in solution quality. Table III ranks the mean improvement per successful move in each move class by problem.[5] Considering the mean improvement in quality from successful moves, *real large* moves produce the greatest improvements in most, but not all, problems (mostly in the 10D instances) while *fake large* moves produce the greatest improvements in four problem instances. *Fake small* moves produce the most consistent improvements, although typically not the highest.

D. Possibility of Misclassification

Given the hard boundary between *small* and *large* moves, it is possible that some moves near or below the threshold may be misclassified: *fake large* moves may result from a similar magnitude but *small* difference vector while *real small* moves may be proportionally very large compared to the difference vector used. In the first case, for most of the problems considered here, less than 6% of *fake large* moves have both difference vector and move within $0.1D$ of the threshold $D_{th} = 0.2D$ (D is the magnitude of the search volume's space-diagonal) and, for all problems, no more than 4% within $0.5D$ of D_{th}. Approximately 18% of *fake large* moves in the 10-dimensional Modified Shekel's Foxholes are, with their difference vector, within $0.1D$ of D_{th}.

Conversely, typically 30% of *small* moves are caused by difference vectors half their magnitude (this proportion is typically smaller for the 10D problems) and approximately 10% are caused by difference vectors one fifth their magnitude. Such moves consequently fit the spirit of the *fake large* classification, if not the requirement that they be *large*. Given that the search landscapes of the test problems are not fractal, distinguishing between these and other *small* moves is unlikely to be informative.

E. Comments on DE/best/1/bin and DE/target-to-best/1/bin

Despite differences in the manner of their search (see Table I), results for DE/best/1/bin and DE/target-to-best/1/bin applied to the test problems exhibit each of the four move classes described in Section II-A.[6] DE/best/1/rand converges very quickly, with some movement of the best solution accompanied by considerable movement of other solutions to its location. Much of this convergence movement is the result of *fake large* moves which, while being less frequent than *real large*, account for more successful *large* moves in all problems except 10D Langerman and Shubert. DE/target-to-best/1/rand exhibits fast convergence on some problems while on problems with numerous local optima it can stagnate with the population fractured in many clusters; its exploration mechanism, which includes a component from the target to the best as well as a random difference vector, is unable to move solutions between the clusters.

[5] *Real small* moves always make the smallest changes.

[6] Full results are not included due to space constraints but may be requested from the author.

IV. CONCLUSIONS

The common understanding of the DE algorithm is that small moves are caused by small difference vectors and large moves by large difference vectors. When the population splits into separate clusters, as can occur in problems with multiple global optima or many competing local optima, movement between these populations was thought to be due to the large difference vectors between members of each cluster. However, small difference vectors applied to solutions in one cluster can produce a new solution that replaces a target from another cluster. Moreover, as the clusters become increasingly tight, *fake large* moves are the *only* mechanism for movement between them: once this point is reached any movement produced by large difference vectors is wasted as the solutions produced will be discarded. Finally, the contribution of such moves is not negligible as they frequently produce improvements in solution quality comparable to genuine large moves.

The knowledge that different mechanisms for large and small moves exist, and are dependent on the current state of the population, may allow for a dynamic DE algorithm that favours (or excludes) certain difference vectors—hence, certain move types—when they are likely to be unproductive.

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] S. Paterlini and T. Krink, "High performance clustering with differential evolution," in *Congress on Evolutionary Computation, CEC 2004*, vol. 2. IEEE, 2004, pp. 2004– 2011.

[3] A. Salman, A. P. Engelbrecht, and M. G. Omran, "Empirical analysis of self-adaptive differential evolution," *European Journal of Operational Research*, vol. 183, pp. 785–804, 2007.

[4] S. Dasgupta, A. Biswas, S. Das, and A. Abraham, "Modeling and analysis of the population dynamics of differential evolution algorithm," *AI Communications - The European Journal on Artificial Intelligence*, 2008.

[5] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *GECCO 06*, Seattle, Washington, USA, 2006, pp. 485–492.

[6] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin: Springer, 2005.

[7] 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.

[8] M. M. Ali, "Differential evolution with preferential crossover," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1137–1147, 2007.