MultiDE: A Simple, Powerful Differential Evolution Algorithm for Finding Multiple Global Optima

Zachary V. Hendershot and Frank W. Moore

Miami University Computer Science and Systems Analysis Department
230 Kreger Hall, Oxford, OH 45056
{henderzv, moorefw}@muohio.edu

Abstract

This paper presents multiDE, an extension of Price and Storn's differential evolution (DE) algorithm that consistently outperforms state-of-the-art search techniques for identifying multiple global optima in multidimensional, discontinuous solution spaces. MultiDE automatically determines appropriate values for control parameters, and periodically updates those values at run-time. MultiDE requires little expert knowledge of the solution space, and is capable of searching both discontinuous and differentiable solution spaces. Innovative use of multiple subpopulations, minimum spanning distances, subpopulation expiration, and precision control contributes to multiDE's speed and effectiveness. Results from several benchmark problems reveal MultiDE's extraordinary power.

Introduction

DE (Price and Storn 1997) is a powerful and efficient technique for optimizing nonlinear and non-differentiable continuous space functions. Its simple yet powerful algorithm is illustrated by Fig. 1. DE was designed as a replacement for traditional methods of solving differential equations, such as simulated annealing (Jones and Forbes 1995) and the well-known simplex algorithm (Nelder and Mead 1965). DE has distinguished itself as a fast and easy-to-use numerical optimization tool.

DE begins by generating a random population of candidate solutions in the form of numerical vectors. The first of these vectors is selected as the target vector. Next, DE builds a trial vector by executing the following sequence of steps:

1. Randomly select two vectors from the current generation.
2. Use these two vectors to compute a difference vector.
3. Multiply the difference vector by weighting factor F (see Fig. 1). It has been found that generally as the number of population members used by DE increases, the value of F should be decreased in order to aid in the convergence process (Storn 1996).
4. Form the new trial vector by adding the weighted difference vector to a third vector randomly selected from the current population.

The trial vector replaces the target vector in the next generation if and only if the trial vector represents a better solution, as indicated by its measured cost value. DE repeats this process for each of the remaining vectors in the current generation. DE then replaces the current generation with the next generation, and continues the evolutionary process over many generations.

DE's shortcomings become apparent when a researcher begins experimenting with problems that have

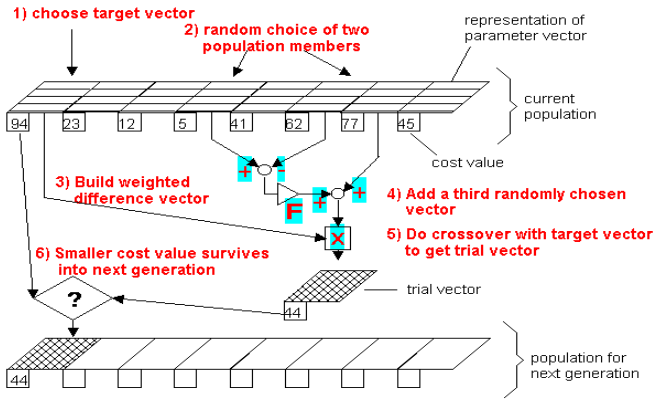

Fig. 1. The Traditional Differential Evolution Algorithm

large, complex solution spaces. Filter design, population are examples of problems that extend traditional assumptions concerning the formation of solutions. In such domains, it may be critical to enhance the problem-solving abilities of scientists and engineers by identifying as many globally optimal solutions as possible.

The New Approach

MultiDE incorporates several enhancements of the original DE framework, resulting in a fast and efficient method of convergence to multiple global optima. Researchers already familiar with DE can migrate to multiDE with minimal effort, and may begin exploiting multiDE's power to determine multiple global optima in complex search spaces.

First, multiDE assigns each candidate solution to one of several simultaneously evolving *subpopulations*. Each subpopulation consists of a user-specified number of vectors. Subpopulations evolve independently from one another, i.e., all three of the vectors used to create each trial vector are randomly selected from the subpopulation containing the corresponding target vector. MultiDE periodically transfers each current global optimum to a separate subpopulation known as "population 0", increases the number of subpopulations, creates a new set of candidate solutions for each subpopulation, and begins the evolutionary process anew. This technique significantly reduces the likelihood of premature convergence to a subset of global optima.

Second, multiDE uses a variable to specify the *precision* required for the current optimization problem. MultiDE considers real-valued solutions that differ by less than this variable's value to be equal. During evolution, each new trial vector is compared to each member of population 0; if the difference between a trial vector and any member of population 0 is less than the specified precision value, then multiDE considers the trial vector to have rediscovered a known globally optimal solution, and the trial vector is dropped from further calculations. This process accelerates the rate at which multiDE converges to new global optima. Reductions in the magnitude of the precision value force multiDE to recognize increasingly smaller differences between candidate solutions.

Third, multiDE dynamically adjusts the *minimum spanning distance* (MSD) between members of different subpopulations. MSD (Rumpler and Moore 2001) is initially calculated to be a factor of the user-specified maximum number of generations. When a new trial vector is created, multiDE computes its Euclidean distance from each of the target vectors in every other subpopulation of the current generation. If this distance is less than the MSD from at least one target vector, multiDE simply moves the trial vector a distance MSD in the opposite direction. This process is repeated until the dynamics, and other applications of differential equations trial vector is at least the MSD from every member of every other subpopulation. By preventing members of multiple subpopulations from gravitating to the same area of attraction, this technique encourages a more thorough search of the solution space, and thus increases the likelihood of finding larger numbers of global optima. Over many generations, multiDE slowly decreases the MSD until its value is less than or equal to the value of the precision variable described above. Each time the number of subpopulations is increased, multiDE resets the MSD to its initial value, and the process is repeated.

Fourth, multiDE introduces *subpopulation expiration* to accelerate the rate of convergence. MultiDE eliminates subpopulations from the currently evolving generation when they fail to discover new global optima after a specified maximum amount of computation. This technique results in a sizable linear increase in convergence speed. Shorter expiring times are most beneficial to researchers who want to quickly identify several solutions, but can forgo the opportunity of finding all possible solutions. A more comprehensive search can be made simply by increasing the expiring subpopulation variable to a much larger value.

MultiDE Solves Classic Optimization Problems

MultiDE's power can best be demonstrated by showing the speed and flexibility with which it solves several well-known multidimensional optimization functions. A "run", as tabulated below, was considered to be a complete execution of the multiDE algorithm with all parameters set and the function to be optimized loaded into the algorithm. Five test runs were made with each function. Each test run used a different random seed. A typical run used 60 vectors per subpopulation. Total computation time was recorded by executing multiDE via the Linux "time" command on a 3.06 GHz Intel Pentium 4 computer running Linux kernel 2.4.22.

The strategies listed below are designated using the standard format that Price and Storn set forth within the source code of their DE algorithm. For example, in DE/best/1/exp, the first value, "DE", indicates that a differential evolution technique has been used. The second parameter specifies the vector to be perturbed during the evolution process: "best" designates the best current population member, while "rand" designates a random population member. The third parameter represents the number of difference vectors to take from the perturbed vector to create a new trial vector. The fourth parameter specifies the crossover method to be used: "exp" designates an exponential method, while "bin" designates a binomial method.

The first equation analyzed was Branin's function (Branin 1972). This continuous function (Fig. 2) has six

global optima within the specified range. DE/rand/1/exp was used as the strategy for optimizing this function. Initial test runs set the expiring subpopulation variable equal to the maximum number of generations. This value resulted in rapid convergence at the expense of a less comprehensive report of possible optimal solutions: the results from five test runs (Fig. 3) indicate that, on average, multiDE found 5.2 unique solutions in an average run time of 7.22 seconds. As few as six subpopulations were used, resulting in an average of 394,385 function evaluations.

Next, the value of the expiring subpopulation variable was increased by a factor of 10, and all five test runs were repeated under otherwise identical conditions. In each run, multiDE located all six global optima (Fig. 4). These results underscore a general rule: to increase the percentage of global optima found, the value of the expiring subpopulation variable must be proportional to the complexity of the solution space and number of solutions desired.

The second equation tested was Shubert's function (Fig. 5). This function has nine global optima within the specified range. Shubert's function lends itself to differential evolutionary optimization because, although its solution space is continuous and differentiable, it also offers sizable slopes that tend to drive adaptive techniques away from the global optima. A DE/rand-to-best/1/exp strategy was used to optimize this function. To increase the likelihood of converging on all possible solutions, the expiring subpopulation variable was increased by a factor of ten. The results of these tests (Fig. 6) illustrate multiDE's ability to rapidly converge to multiple optima. All five test runs determined all nine global optima solutions in an average execution time of only 5.47 seconds.

The third equation used to test multiDE was Rosenbrock's saddle (Rosenbrock 1960), shown in Fig. 7. This equation is an exception in this test suite: it has only one optimal solution. The reasons for its inclusion are to allow comparisons with more conventional optimizers, and to show that multiDE is flexible enough to solve problems having a single global optimum.

Using parameters similar to those used in previous tests and assuming that $n = 2$, multiDE quickly found the global optimum, in spite of the sizable overhead of multiDE's multi-pass algorithm. Note that after finding the single solution, multiDE continued executing many more iterations in an attempt to find additional solutions by increasing the number of subpopulations. Waiting for these additional subpopulations to complete introduced a function evaluation penalty that increased the total number of function evaluations (Fig. 8).

An interesting phenomenon arises when multiDE is applied to the relatively simple equation illustrated in Fig.

$$\textit{Minimize} \quad f(x) = \left(x_2 - \left(\frac{5}{4\pi^2}\right)x_1^2 + \left(\frac{5}{\pi}\right)x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_i) + 10$$

$$\textit{Subject To} \quad -5 \le x_1, x_2 \le 15$$

Fig. 2. Branin's Function

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	5	8	782710	11.342	DE/rand/1/exp
2	5	6	233580	6.57	DE/rand/1/exp
3	5	6	227943	4.666	DE/rand/1/exp
4	5	7	501366	9.984	DE/rand/1/exp
5	6	6	226325	3.587	DE/rand/1/exp
Average	5.2	6.6	394384.8	7.2298	

Fig. 3. Branin's Function Test Results: Shorter Subpopulation Expiration Times

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	6	9	543094	41.190	DE/rand/1/exp
2	6	10	403493	36.243	DE/rand/1/exp
3	6	9	603912	42.302	DE/rand/1/exp
4	6	9	430934	37.584	DE/rand/1/exp
5	6	10	574833	44.238	DE/rand/1/exp
Average	6.0	9.4	511253.2	40.3114	

Fig. 4. Branin's Function Test Results: Longer Subpopulation Expiration Times

9. This equation has a total of sixteen global optima. When run with the same parameters specified for Shubert's function, multiDE quickly found up to ten unique solutions, but after multiple iterations failed to find the remaining solutions. Fig. 10 shows these results over five independent test runs.

To allow multiDE to solve problems having a larger number of global optima, the expiring subpopulation variable must be increased. After the expiring variable was increased by a factor of ten, multiDE used 22 subpopulations and 9,412,512 function evaluations to converge on all sixteen solutions in a single run in 15.72 seconds (Fig. 11).

MultiDE Outperforms Classic Optimization Algorithms

Efstratiadis (Efstratiadis 2001) defines effectiveness as follows:

> *[Effectiveness] indicated the probability of finding a global optimum starting from any random initial solution (or population of solutions)... A measure of the effectiveness of an algorithm in a specified problem is the number of successes out of a predefined number of independent runs.*

To demonstrate its effectiveness, multiDE was compared to the following classic algorithms: a multistart simplex technique (Torn and Zhilinskas 1989); a genetic algorithm-based method (Goldberg 1989); a shuffled complex evolution method (Duan, Gupta, and Sorooshian 1993); and a powerful annealing-simplex algorithm (Efstratiadis 2001) inspired by simulated annealing. Each algorithm was tested using Griewank's function and

$$\textit{Minimize} \quad f(x) = \left(\sum_{j=1}^{5} j\cos\left((j+1)x_1 + j\right)\right)\left(\sum_{j=1}^{5} j\cos\left((j+1)x_2 + j\right)\right)$$
$$\textit{Subject To} \quad -10 \le x_i \le 10 \quad \textit{for} \quad i = 1,2$$

Fig. 5. Shubert's Function

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	9	7	304930	4.546	DE/rand-to-best/1/exp
2	9	9	489445	6.549	DE/rand-to-best/1/exp
3	9	7	387545	5.528	DE/rand-to-best/1/exp
4	9	8	419430	5.762	DE/rand-to-best/1/exp
5	9	7	334309	4.984	DE/rand-to-best/1/exp
Average	9.0	7.6	387131.8	5.4738	

Fig. 6. Shubert's Function Test Results

$$\textit{Minimize} \quad f(x) = \sum_{i=1}^{n/2} 100\left(x_{2i} - x_{2i-1}^2\right)^2 + \left(1 - x_{2i-1}\right)^2$$
$$\textit{Subject To} \quad -10 \le x_i \le 10 \quad \textit{for } i = 1,\ldots., n.$$

Fig. 7. Rosenbrock's Saddle

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	1	7	449231	1.221	DE/rand-to-best/1/exp
2	1	7	449231	1.22	DE/rand-to-best/1/exp
3	1	7	449231	1.214	DE/rand-to-best/1/exp
4	1	7	453803	1.231	DE/rand-to-best/1/exp
5	1	7	465712	1.237	DE/rand-to-best/1/exp
Average	1	7	453441.6	1.2246	

Fig. 8. Rosenbrock's Saddle Test Results

$$\textit{Minimize} \quad f(x) = \left|(x-3)*(x-6)*(x-9)*(x-12)\right| + \left|(y-4)*(y-8)*(y-12)*(y-16)\right|$$
$$\textit{Subject To} \quad -20 \le x, y \le 20$$

Fig. 9. Simple Test Function with 16 Global Optima

Michalewicz's function. The calculated effectiveness of each function is defined as the percentage of global optima found.

Griewank's function (Fig. 12), as tested, has a ten-dimensional solution space of sufficient complexity to warrant extended convergence times and an increased number of function evaluations. This function is reported to have over 1000 optima in the range of interest (Efstratiadis 2001), and therefore presents an interesting benchmark for multiple global optima algorithms. Despite the multi-dimensional, non-convex nature of this function, the effectiveness of most of the algorithms in our test suite approached 100 (Fig. 14). MultiDE required only 7.35 seconds to find 100% of the solutions.

Michalewicz's function is illustrated in Fig. 13. Efstratiadis (Efstratiadis 2001) states that this function has more than 100 global optima in the specified range. This function, as tested, has a two dimensional solution space that proved to be very difficult for traditional optimization techniques: for the traditional four algorithms tested, the highest observed effectiveness rating was 58 (Fig. 14). In contrast, multiDE's effectiveness in solving Michalewicz's function was 96.

The importance of this finding is that, although the overhead of our multiple-pass evolutionary approach may be significant for simple test problems, multiDE's comprehensive search capabilities allow it to quickly converge on multiple globally optimal solutions in solution spaces that are too complex for the traditional optimization techniques included in this test suite. This result highlights the most important property of multiDE: its power to solve complex optimization problems quickly and reliably. MultiDE automatically adapts to the complexity of the given solution space, effectively utilizing the power of multiple simultaneously evolving subpopulations to find arbitrarily large numbers of global optima with less computational cost than traditional search techniques.

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	8	15	4147023	7.052	DE/rand-to-best/1/exp
2	10	13	3011941	5.443	DE/rand-to-best/1/exp
3	9	13	2999248	5.409	DE/rand-to-best/1/exp
4	9	13	3012578	5.442	DE/rand-to-best/1/exp
5	10	16	4762764	7.631	DE/rand-to-best/1/exp
Average	9.2	14	3586710.8	6.1954	

Fig. 10. Simple Multiple Optima – Results for Short Test Runs

Run #	# of Unique Solutions Found	Maximum # of Subpopulations	NFE	Time	Strategy
1	16	22	9412512	15.722	DE/rand-to-best/1/exp

Fig. 11. Simple Multiple Optima – Results for a Longer Test Run

$$\textit{Minimize} \quad f(x) = \frac{1}{4000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$
$$\textit{Subject To} \quad -600 \le x_i \le 600$$

Fig. 12. Griewank's Function (Griewank 1981)

$$\textit{Minimize} \quad f(x) = -\sum \sin(x_i) * \sin^{20}$$
$$\textit{Subject To} \quad 0 \le x_i \le \pi$$

Fig. 13. Michalewicz's Function (Sun and Laio 2001)

Test Function	Dimensions (*n*)	Number of Optima	Multistart Simplex	Genetic Algorithm	SCE-UA	Annealing-simplex	multiDE
Griewank	10	>1000	100	89	100	99	100
Michalewicz	2	>100	35	31	44	58	96

Fig. 14. Test Results for Complex Functions: Effectiveness (%)

MultiDE was designed to be easy to use. Very few adjustments to control parameter values were necessary in order for the algorithm to reliably converge on multiple global optima; for example, to transition from testing Griewank's function to testing Michalewicz's function, the only values modified in the configuration file to reliably converge on multiple global optima were number of variables generated and the DE evolution strategy. (It is possible to change a larger number of variables within the configuration file, if the researcher desires finer control over the optimization process.) The nature of the algorithm lends itself to less reliance on the values of control parameters and more flexibility in convergence.

Conclusions

This paper described a new differential evolution algorithm that represents a significant contribution to the state-of-the-art in numerical optimization. Experimental results demonstrated multiDE's ability to consistently identify multiple global optima for complex functions (such as Michalewicz's function) that prove to be troublesome for such traditional optimization techniques as the multistart simplex method, the standard binary encoded genetic algorithm, the shuffled complex evolution method, and the annealing-simplex method. MultiDE allows researchers to experiment with a simple, flexible, and powerful differential evolution solver without a steep learning curve. MultiDE can be applied to a broad range of challenging optimization problems. Future research efforts will exploit multiDE's power to solve real-world problems with arbitrarily complex solution spaces.

References

Branin, F. H. Jr., 1972. "Widely convergent method for finding multiple solutions of simultaneous nonlinear equations", *IBM J. of Research and Development* 16(5): 504-522.

Duan, Q., V. Gupta, and S. Sorooshian, 1993. "A Shuffled Complex Evolution Approach for Effective and Efficient Global Minimization", *J. of Optimization Theory and Applications* 76(3): 501-521.

Efstratiadis, A., 2001. *Investigation of global optimum seeking methods in water resources problems*, M. Sc. thesis, Department of Water Resources, Hydraulic and Maritime Engineering, National Technical University of Athens.

Goldberg, D., 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.

Griewank, A., 1981. "Generalized Descent for Global Optimization", *J. of Optimization Theory and Applications* 34: 11-39.

Jones, A. and G. Forbes, 1995. "An Adaptive Simulated Annealing Algorithm for Global Optimization over Continuous Variables", *J. of Global Optimization* 6(1): 1-37.

Michalewicz, Z. and T. Logan, 1994. "Evolutionary Operator for Continuous Convex Parameter Spaces", *Proc. 3rd Annual Conf. on Evolutionary Programming*, 84-97, World Scientific Publishing.

Nelder, J. and R. Mead, 1965. "A Simplex Method for Function Optimization", *Computer J.* 20(1): 84-85.

Price, K. and R. Storn, 1997. "Differential Evolution: Numerical Optimization Made Easy", *Dr. Dobb's J.*, April 1997, 18-24.

Rosenbrock H., 1960. "An Automatic Method for Finding the Greatest or Least Value of a Function", *Computer J.* 3: 175-184.

Rumpler, J. and F. Moore, 2001. "Automatic Selection of Sub-populations and Minimum Spanning Distances for Improved Numerical Optimization", *Proc., Congress on Evolutionary Computation 2001* 1: 38-43, IEEE.

Storn, R., 1996. "Differential Evolution Design of an IIR-Filter", *Proc. IEEE International Conf. on Evolutionary Computation*, Nagoya, Japan, 268-273, IEEE.

Sun, C-T. and Y-H. Liao, 2001. "An Educational Genetic Algorithm Learning Tool", *IEEE Trans. on Education* 44(2), IEEE.

Torn, A. and A. Zhilinskas, 1989. "Global Optimization", *Lecture Notes in Computer Science* 350, Springer-Verlag.