DSE: A Hybrid Evolutionary Algorithm with Mathematical Search Methods

Francisco Viveros Jiménez

Universidad del Istmo Campus Ixtepec, Ciudad Universitaria S/N, Cd. Ixtepec, Oaxaca, México. fviveros@bianni.unistmo.edu.mx

Abstract. In this article a new technique Deep Sea Exploration, termed DSE, is proposed. DSE is a hybrid mathematical and evolutionary method used for numerical optimization. DSE uses evolutionary methods, such as three-individual crossover (taken from Differential Evolution), Gaussian random mutation (also proposed in this article) and a traditional search method for exploitation (similar to Hooke & Jeeves' search method) and adaptive behavior. DSE is applied to a set of benchmark functions and compared with some evolutionary techniques to show its efficiency.

1. Introduction

The numerical optimization problem is unsolved because of its high level of complexity. Many approaches have been deployed in recent years; the most popular are the stochastic and mathematical techniques. Mathematical methods can find an optimal value with a high level of accuracy, but in complex search spaces it is difficult to find the global optimum, and on many occasions it may not be possible. In addition, these methods require a lot of processing power. Stochastic methods are more efficient in exploring complex search spaces and they require less processing power than mathematical methods. However, stochastic methods do not guarantee finding the global optimal value.

Evolutionary algorithms are very efficient stochastic techniques. Evolution strategies (ES) were developed in Germany by Schwefel [8] and Rechemberg [9]. ESs imitate natural (organic) evolution. The number of parents (μ) is defined by the user. The ESs use multiple parents to create offspring [10]. This process is then repeated to create offspring (λ individuals) which are defined by the user. Once all offspring are created, mutation (Gaussian perturbation) is applied. In the (μ+λ)-ES, the mutants and the parent population become one and selection is based on the ranking of individuals' fitness. The λ best individuals will form the next generation of parents. This process is repeated for n generations. ESs perform well and therefore DSE implements some concepts from the (μ+λ)-ES in order to ensure successful exploration.

Hooke & Jeeves' (HJ) Direct Search method [7] is a mathematical method. HJ is used to find the optimal value for functions of n variables. The HJ algorithm sequence is:

1. Take an initial search point X0 and copy it in XB.

© G. Sidorov, B. Cruz, M. Martínez, S. Torres. (Eds.)
Advances in Computer Science and Engineering.
Research in Computing Science 34, 2008, pp. 59-67

Received 19/03/08
Accepted 26/04/08
Final version 03/05/08

2. HJ explores from XB with an initial step length of D, in the first variable in both directions.
3. If the exploration points decrease, the function-1 value (in the case of minimization) XB will move in that direction.
4. Repeat the complete process with the rest of the variables.
5. When completed, the new value of F is evaluated.
6. If F decreases, move in the diagonal formed between X0 and XB. The new point will become the new X0 and XB. If F increases, D is diminished by half.
7. Repeat the entire process until D reaches a user-defined error value.

Figure 1 illustrates the process.

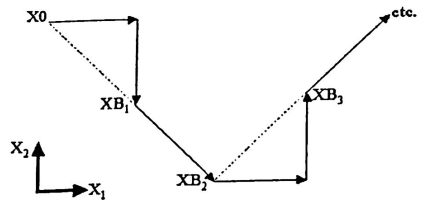

Fig. 1. Hooke&Jeeves General Idea.

DSE implements a method very similar to HJ to perform a successful exploitation. This article describes a hybrid technique, which works in a similar way to $(\mu+\lambda)$-ES and implements a numerical method for increasing performance and also implements adaptive parameters and some concepts from genetic algorithms (GAs).

2. DSE Algorithms

DSE is based on the behavior of a group of human sea explorers. DSE has 3 parameters:

1. B, which represents the number of biologists whose primary objective is to explore the deepest sea bed.
2. E, which represents the number of experienced explorers who assist this biologist.
3. N, which represents the number of iterations (generations).

The explorers and the biologist are being guided by the wise, old ship's captain. First, the captain tells the biologist to explore a random point of the

sea bed. Next, he decides to send a number of explorers to new points, and the rest to search in close proximity to the biologist's points. Then, the explorers dive into the sea at their assigned point and descend to the deepest possible point. Once each explorer reaches a point they judge to be the deepest, they report the success of their mission and return to the ship.

Then, the captain evaluates the new points and sends the B biologist to the deepest point of the combination of explorers' and biologist's points. At the next time interval (generation in evolutionary algorithms terms) the captain takes the results of the most recent exploration and examines it. If the explorers find a new deepest point the captain decides to send more explorers to new points, however, if the opposite is the case the captain decides to send more explorers to search in close proximity to the biologist's points. This process repeats n generations.

Figure 2 shows the DSE process in evolutionary terms.

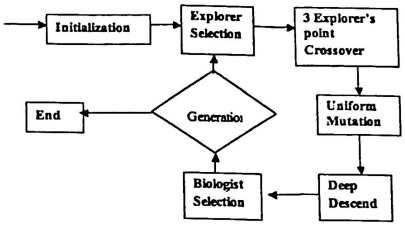

Fig. 2. DSE process.

DSE process in evolutionary terms is:

1. Initialize B random biologist and evaluate all. The biologist is equivalent to the μ parents in the ESs. The captain should be initialized as 0, which means that all explorers are going to start at random points. Take F value from the first biologist as the last value of F (UFX).
2. Compare UFX with the F value from the biologist. If F is better than UFX the captain's value decreases by 5% (the minimal is 1). If the opposite is the case the captain's value increases by 10% (the max is E-1).
3. Initialize the captain's explorers together with the biologist and the rest with random points. Make UFX equal to the F value of the first biologist.
4. Perform a Gaussian random mutation. This mutation consists of a slight alteration in a random number of variables from the first explorer. The number of variables needs to be generated with a normal distribution with a mean of 0 and a standard deviation of 1. The formula for mutation is:

$$x_j = x_j + x_j \quad rndreal(-0.01, 0.01) \tag{1}$$

In equation 1, j needs to be randomly generated.

5. Perform a three-individual (three-membered) crossover on the first captain's explorers. The formula for this is:

$$x_j^a = (x_j^b + x_j^c + x_j^d)/3 \tag{2}$$

Where b=rnd(1,captain), c and d are equal to rnd(1,E) all of them are individuals from the explorer set, and j represents the variable number.

6. Do a deep descent for all the explorers. To do this initialize D (step size) with 2*P (P is defined by the user and is also the minimal step). Then move the individual as in HJ with two changes: do not do the diagonal movement and D will grow if the Function value decreases (minimization case). Equations 3, 4 and 5 show how to increase the value of D.

$$FD = |(F_i - F_{i-1})/D| \tag{3}$$

$$FDD = |(FD_i - FD_{i-1})/D| \tag{4}$$

$$D = |(FD/FDD| \tag{5}$$

Where the initial values are:

$$FD_0 = D_0, FDD_0 = 1.$$

7. Create a list with the results of the new E explorer's points and the B Biologist's points. Sort the list and make the B biologist equal to the first B points from the sorted list.
8. If the generation limit is reached finish and report the results, otherwise return to step 2.

The Biologist represents the population. The explorer represents the offspring. The captain represents the adaptive quality.

3. Results and Discussions

Each test case ran for T=2000 generations and all the cases were run 30 times each. The parameters were set in the following form: B=100, E=100, P=0.000001. This set of parameters is not the optimal for each function. These paramaters were chosen only for experimental purposes. All functions are unconstrained and are specified in appendix A at the end of this article. The benchmark functions were taken from [11]. All these functions have 0 for optimal value.

Results in Table 1 show that DSE performs well in most benchmark functions. DSE is consistent because the standard deviation is small in most cases and results are close to the optimal value.

Figure 3 displays the convergency graphs for a random test displayed in some benchmark functions. These graphs show that DSE finds the global optimal zone in 1000 generations in the majority of cases. Also, the graphs show that DSE exploitation is relatively slow, because DSE does not find the optimal value with the P precision expected, but it returned a close value in most cases.

Table 1. Results obtainend on test runs

	F_1	F_3	F_5	F_7	F_9	F_{10}	F_{11}
Best	0.15267	18.96260	9.03024	0.00078	0.06207	2.05194	0.29521
Worst	0.72645	75.18479	106.79471	0.00881	4.37922	4.09525	0.69629
Mean	0.28892	43.61514	34.97672	0.00368	1.50967	2.93939	0.49720
Standard deviation	0.12287	13.19212	25.22960	0.00193	1.27074	0.49897	0.10922

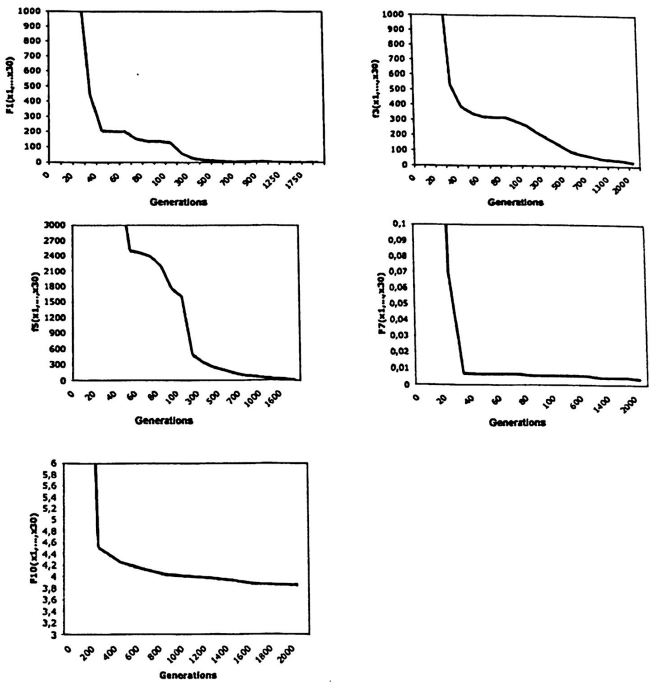

Fig. 3. Convergency graphs for test runs.

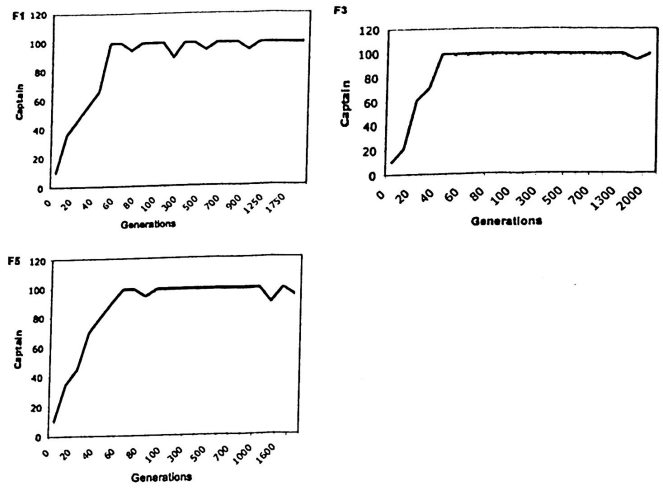

Fig. 4. Captain Behavior for functions: F_1, F_3 and F_5.

Figure 4 shows graphs for the adaptive behavior. These graphs show that the captain stimulates exploration in the early generations and encourages exploitation in the final stages.

Table 2 shows a comparison between a (100+100)-ES and DSE. Table 3 shows the comparison between the mean values from DSE and some Diferential Evolution (DE) variants. The results from DE variants were taken from [11]. Table 4 shows the comparison between the mean values from DSE and some Particle Swarm Optimization (PSO) variants. The results from FPSO [6], DPSO [4], DEPSO were taken from [2]. Best results are marked in boldface.

Table 2. Comparative table of performance for F_{10}

F_{10}	*DSE*	*ES*
Best	**2.051**	13.4
Worst	**4.095**	15.07
Mean	**2.939**	14.44
Standar deviation	**0.498**	0.526

Table 3. Comparative table of performance between DSE and DE variants

	F_1	F_3	F_5	F_7	F_9	F_{10}	F_{11}
DSE	0.28892	43.61514	34.97672	0.00368	1.50967	2.93939	0.49720
rand/1/bin	**0.0**	0.02430	19.57789	**0.0**	**0.0**	**0.0**	0.00111
rand/1/exp	**0.0**	**0.0**	**6.69606**	**0.0**	97.75393	0.08003	0.00007
best/1/bin	**0.0**	**0.0**	30.39087	**0.0**	**0.0**	**0.0**	**0.00072**
best/1/exp	407.972	10.6078	132621	0.070545	40.00397	9.3961	5.9278

Table 4. Comparative table of performance between DSE and PSO variants

	F_5	F_9	F_{11}
DSE	**34.9767**	1.5096	0.4972
DPSO	57.2802	4.2265	0.0119
DEPSO	60.6405	**0.0**	**0.0055**
FPSO	124.4184	22.5239	0.0149

DSE is competitive when compared with other techniques. It is also performs better in some cases.

4. Conclusions

This article describes a hybrid mathematical and evolutionary method called DSE. The hybrid strategies provide the exploration using techniques from ESs and GAs. The exploitation is provided by a numerical technique similar to HJ. The test shows that DSE performs well in most test cases. However, more comparative work and studies should be carried out to provide a fuller, more detailed analysis and refinement. The adaptive behavior performs in the desired way and this is shown in the performance of DSE. Some unreported tests show that DSE without the adaptive behavior perform less efficiently. Future work with constrained function should be performed to observe the behavior of DSE.

Acknowledgments

Thanks to Damhait Dennis for the grammar check, to Ignacio Luna Espinoza for the statistical test check and to UNISTMO for the opportunity. I also want to thank Karla Arroyo Martello for all the support that she gives me.

References

1. Beyer, H-G., Schwefel, H-P.: Evolution strategies: a comprehensive introduction. Natural Computing, Volume 1, Number 1, Springer Netherlands. (2002) 3–52.
2. Wen-Jun, Z., Xiao-Feng, X.: DEPSO: Hybrid Particle Swarm with Differential Evolution Operator. IEEE Int. Conf. on Systems, Man & Cybernetics. (2003) 3816–3821
3. Storn, R., Price, K.: Differential Evolution - a simple and efficient heuristic for global optimization. Journal of Global Optimization, Volume 11, Number 4, Springer Netherlands. (1997) 341–359.
4. Shi, Y H., Eberhart, R C.: Fuzzy adaptive particle swarm optimization. IEEE Int, Conf. on Evolutionary Computation. (2001) 101–106.
5. Wolpert, D H., Macready W G.: No free lunch theorems for optimization. IEEE Trans. on Evolutionary Computation. (1997) 67–82.
6. Xie, X F., Zhang, Z L.: A dissipative particle swarm optimization. Congress on Evolutionary Computation. (2002) 1456–1461.
7. Hooke, R., Jeeves, T A.: Direct Search Solution of Numerical and Statistical Problems. Journal of the ACM, Volume 8. (1961) 212–229.
8. Schwefel, H P.: Kybernetische evolution als strategie der experimentellen forschung in der strmungstechnik. Diploma thesis, Technical Univ. of Berlin. 965W. Atmar, Eds. La Jolla, C Evolutionary Programming Society. (1992) 35–42.
9. Rechenberg, I: Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biolgischen evolution. Stuttgart: Frommann- Holzboog Verlag. the Sec. Parallel Problem Solvingfvom Nature Conf., R. Manner and B. Manderick, Eds. The Netherlands: Elsevier Science Press. (1973) 175–186.
10. Schwefel, H P.: Numerische optimierung von computer-modellen mittels der evolutionsstrategie. Interdisciplinary systems research, volume 26 Basel: Birkhuser. (1997).
11. Mezura-Montes, E., Coello, C CA., Velazquez, R J.: A comparative study of differential evolution variants for global optimization. Proceedings of the 8th annual conference on Genetic and evolutionary computation. (2006) 485–492.

APPENDIX

A Test Functions

The benchmark functions were take from [11].

$f01$ - Sphere Model

$$f_1(x) = \sum_{i=1}^{30} x_i^2$$
$$-100 \leq x_i \leq 100$$
$$min(f_1) = f_1(0, ..., 0) = 0$$

$f03$ - Schwefel's Problem 1.2

$$f_3(x) = \sum_{i=1}^{30} \left(\sum_{j=1}^{i} x_j \right)^2$$
$$-100 \leq x_i \leq 100$$
$$min(f_3) = f_3(0, ..., 0) = 0$$

$f05$ - Generalized Rosenbrock's Function

$$f_5(x) = \sum_{i=1}^{29} |100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2|$$
$$-30 \leq x_i \leq 30$$
$$min(f_5) = f_5(0, ..., 0) = 0$$

$f07$ - Quartic Function with Noise

$$f_7(x) = \sum_{i=1}^{30} i x_i^4 + random[0, 1)$$
$$-1.28 \leq x_i \leq 1.28$$
$$min(f_7) = f_7(0, ..., 0) = 0$$

$f09$ - Generalized Rastrigin's Function

$$f_9(x) = \sum_{i=1}^{30} [x_i^2 - 100cos(2\pi x_i) + 10]$$
$$-5.12 \leq x_i \leq 5.12$$
$$min(f_9) = f_9(0, ..., 0) = 0$$

$f10$ - Ackley's Function

$$f_{10}(x) = -20exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2}\right) - exp\left(\frac{1}{30}\sum_{i=1}^{30} cos(2\pi x_i)\right) + 20 + e$$
$$-32 \leq x_i \leq 32$$
$$min(f_{10}) = f_{10}(0, ..., 0) = 0$$

$f11$ - Generalized Griewank's Function

$$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$
$$-600 \leq x_i \leq 600$$
$$min(f_{11}) = f_{11}(0, ..., 0) = 0$$