Evonorm: Easy and Effective Implementation of Estimation of Distribution Algorithms

Luis Torres-Trevino

Corporación Méxicana de Investigación en Materiales, Oceanía 190,
Tel. (52) 844-411-32-00, ext 1251,
25290 Saltillo, Coahuila, México
ltorres@comimsa.com.mx
http://www.comimsa.com.mx

Abstract. This paper shows some preliminary results about the performance of an estimation of distribution algorithm of easy implementation but effective to solve optimization problems. Computer simulation compares Evonorm versus Evolution Strategies to optimize complex functions. The results show a better efficiency and easy implementation than Evolution Strategies in the optimization of De Jong functions.

1 Introduction

There is a new tendency to generate simplified versions of evolutionary algorithms where crossover and mutation procedures are replaced. The population is built by a model that represents an estimation of distributions of the best individuals selected. Evonorm is a new evolutionary algorithm where the population is built by random variables with normal distribution. The parameters of these random variables are determined by the calculation of the mean and the standard deviation of selected population of solutions. The evolutionary algorithm replaces the crossover and the mutation procedure with new procedures to calculate parameters of random variables and to generate new individuals from these random variables with normal distribution.

Evonorm is an easy and effective way to apply estimation of distribution algorithms (EDAs). Evonorm can be used to find the decision variables that optimize a given function. There are several approaches to apply EDAs and can be classified in two classes, discrete and continuous. Examples of discrete EDAs are PBIL [1], UMDA [2], and CGA [3]. Examples of continuous EDAs are UMDAc [4] and IDEA [5]. In every class there are two subclasses, EDAS that consider a dependences between decision variables like BOA [6], hBOA[7], and eCGA[8], and EDAs that consider a independencies between variables. In continuous EDAs there is the same sub classification [9].

EDAs replace the use of crossover and mutation mechanism so it is expected a simplification in the implementation of an EDA, but in some of them it implies the use of high cost search mechanism to get an effective model to estimate the distribu-

© A. Gelbukh, S. Suárez. (Eds.)
Advances in Computer Science and Engineering.
Research in Computing Science 23, 2006, pp. 75-83

Received 09/08/06
Accepted 03/10/06
Final version 13/10/06

tion of individuals selected in order to generate a new population. It is like a new low-level optimization problem into high-level optimization problem. Some experiments have shown the efficient of EDAs versus another evolutionary algorithms and the conclusion is the same, the EDAs requires more computational time and do not improve significantly the solutions found [10]. Evonorm can be effective founding good solutions but without the use of complex search mechanism to estimate distributions.

The present section is the introduction. The second section is an introduction of EvoNorm. The third section shows the performance of EvoNorm versus Evolution strategies (μ, λ) to make comparison between them optimizing De Jong Functions. The conclusion and future work is given in section fourth.

2 EVOlutionary Algorithm of Random Variables with NORMal Distributions (Evonorm)

Evonorm uses random variables with normal distribution. The normal distribution function is a random variable and describes many random phenomena that occur in every day life. It is simulated the normal distribution function with two parameters, the first is the mean and it is a numeric measure of the central tendency of the random variable. The second parameter is the standard deviation and it is a measure of the dispersion of a variable around the mean. A normal distribution function can be used to represent a set of possible values of a decision variable, so it is necessary to use a set of parameters (mean and standard deviation) of the normal distribution function per decision variable. Equation 1 shows an easy implementation of a random variable with a normal distribution.

$$N(\mu, \sigma) = \mu + \sigma \sum_{i=1}^{12} U \qquad (1)$$

Where: μ is the mean, σ is the standard deviation, and U is a uniform random number generator. The Evonorm procedure has the same philosophy of an evolutionary algorithm because there are an evaluation process, a selection, and a variation procedure where the crossover and mutation are substituted by new procedures, the calculation of the parameters of the normal distribution functions per decision variable and the generation of a new population.

In continuous optimization a vector of real numbers can represent continuous decision variables. It is used a random variable per decision one. EvoNorm evolves the random variables to generate new real vectors of decision variables. These variables will be evaluated and the best of them will be selected to calculate new parameters of the distribution function to generate a new population. The process is repeated again and again (Table I). The calculation of the mean and standard deviation is a common known arithmetic procedure (2, 3)

$$\mu(pr) = \sum_{i=1}^{n} PS(pr,i)/n \tag{2}$$

$$\sigma(pr) = \sqrt{\sum_{i=1}^{n} (PS(pr,i) - \mu(pr))^2} \tag{3}$$

Where pr represents the decision variables involved, and PS represents the selected individuals. n represents the number of individuals selected. PS is a matrix of $NT\Pr$ columns and NTR rows both constants represents the total of parameters and individuals respectively.

Table 1. Seudocode of evonorm

```
1) Generation of a uniform random popula-
   tion P of size m.
2) Evaluation of the m individuals.
3) Selection of the best n individuals
   (n<m)
4) Calculation of mean and standard de-
   viation from n selected individuals.
5) Modify standard deviation if intensive
   exploration is active.
6) Generation of a new population of size
   m from random variables with parame-
   ters calculated in (4) and (5)
7) If a criterion satisfied then end else
   go to step (2)
```

EVONORM is similar to UMDAc because uses random variables that generate numbers with Gaussian distribution. The parameters of these random variables are calculated from the population selected too. Evonorm searches the appropriate parameters of the normal distribution to improve the random variables associated.

Evonorm consider an intensive exploration phase because some functions are difficult to optimize. In this phase it is used a constant standard deviation equivalent to the half of the range of a decision variable and it is used on a half of the total generations of Evonorm. Some functions do not require the intensive exploration phase, nevertheless the use of this procedure improve lightly the performance of the search in these functions.

The exploration is implemented by a simple condition: If the generation counter is below of the 50 percent of the total generations then the standard deviation is equal to the half of the range of the decision variable (to get a very intensive exploration) else Evonorm uses the standard deviation calculated from the selected population. The calculated mean is always used.

3 Performance Comparisons between Evonorm and Evolution Strategies

It is analyzed the performance of Evonorm and Evolution Strategies (μ, λ) without recombination [11] on classic test De Jong functions [12, 13]. These functions take care to include continuity, discontinuity, convex, nonconvex, unimodal, multimodal, quadratic, non quadratic, low dimensional, high dimensional, stochastic and deterministic characteristics (4 to 8)

The exploration is applied during the 50 percent of the total generations at the beginning of the run. Table II shows the Evonorm parameters used to optimize every function. Evolution Strategies was adjusted to get the same number of evaluations. The average performance of both algorithms in 100 runs to optimize all De Jong functions are shown from figure 1 to figure 5.

De Jong function 1

$$f(X) = \sum_{i=1}^{3} x_i^2 \tag{4}$$

Where $-5.12 \leq x_i \leq 5.12$

De Jong function 2

$$f(X) = 100(x_1^2 - x_2) + (1 - x_1)^2 \tag{5}$$

Where $-2.048 \leq x_i \leq 2.048$

De Jong function 3

$$f(X) = \sum_{i=1}^{5} \mathrm{int}(x_i) \tag{6}$$

Where $-5.12 \leq x_i \leq 5.12$

De Jong function 4

$$f(X) = \sum_{i=1}^{30} i x_i^4 + gauss(0,1) \tag{7}$$

Where $-1.28 \leq x_i \leq 1.28$

De Jong function 5

$$f(X) = \frac{1}{1/K + \sum_{j=1}^{25} f_j^{-1}(x1, x2)} \tag{8}$$

Where $f(x1, x2) = c_j + \sum_{i=1}^{2} (x_i - a_{ij})^6$

Where $-65.536 \le x_i \le 65.536$, $K = 500$, $c_j = j$, and

$$[a_{ij}] = \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & \dots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -32 & -16 & \dots & 32 & 32 & 32 \end{bmatrix}$$

Table 2. Evonorm parameters used to optimize every De Jong function

	De Jong 1	De Jong 2	De Jong 3	De Jong 4	De Jong 5
Generations	50	100	50	100	150
Total of individuals	25	50	50	50	200
Individuals selected	5	10	10	10	40
Use exploration	Not	Yes	not	not	Yes

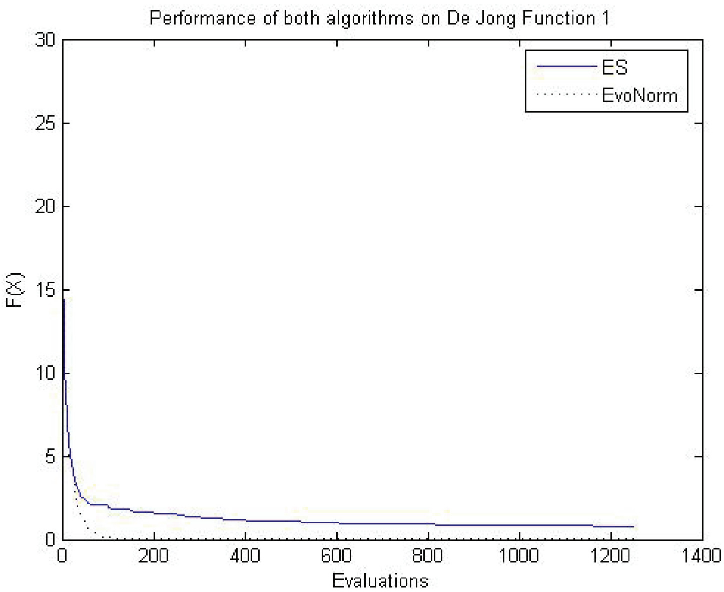

Figure 1. Performance of the algorithms to optimize De Jong function 1.

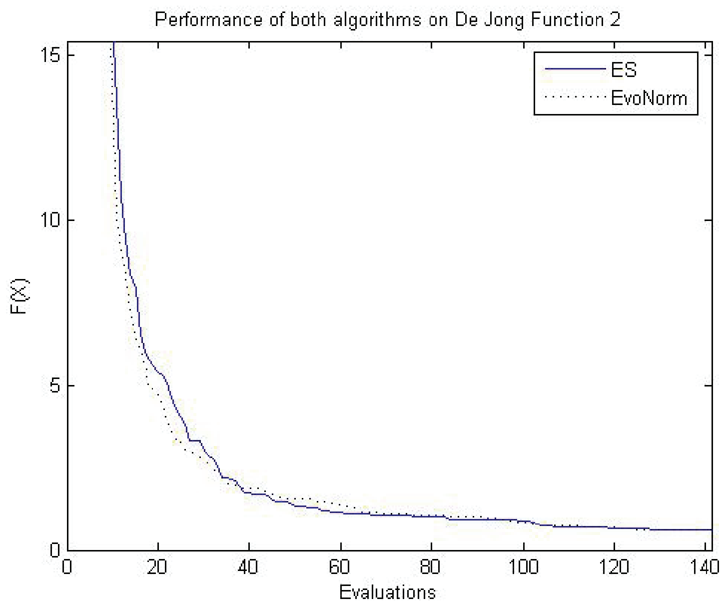

Figure 2. Performance of the algorithms to optimize De Jong function 2.

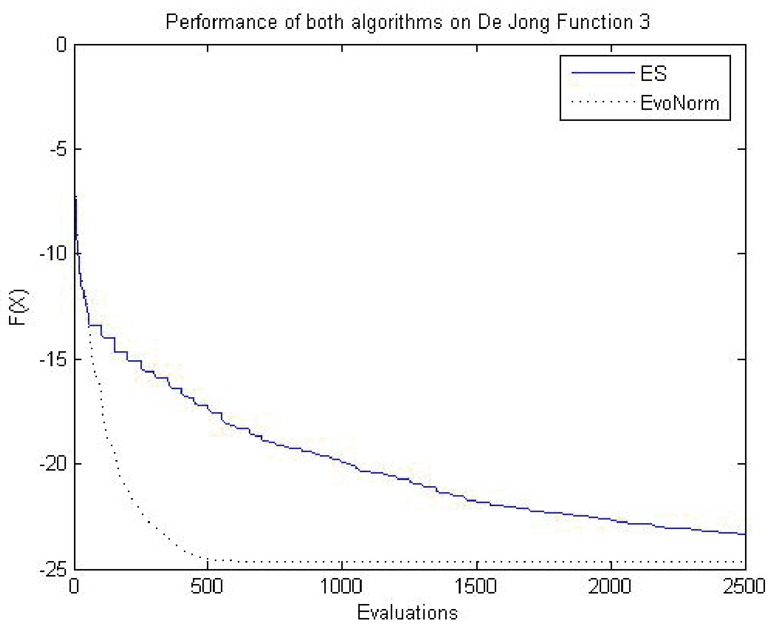

Figure 3. Performance of the algorithms to optimize De Jong function 3.

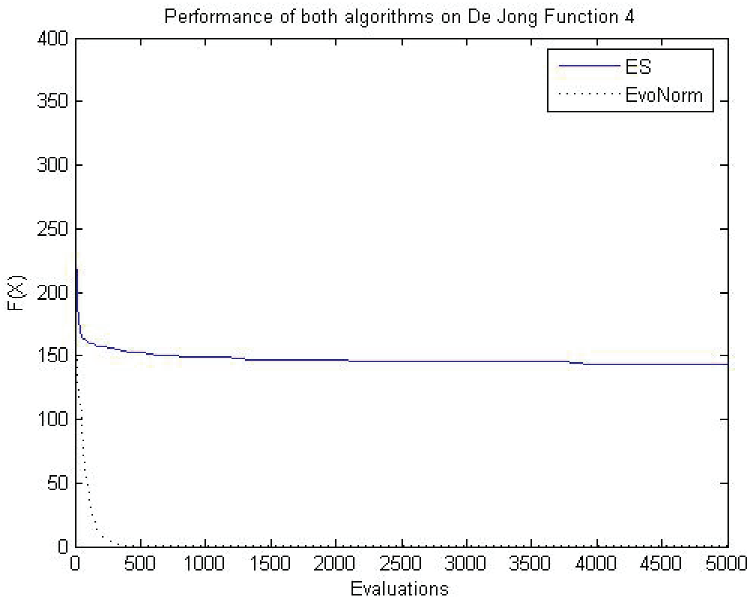

Figure 4. Performance of the algorithms to optimize De Jong function 4.

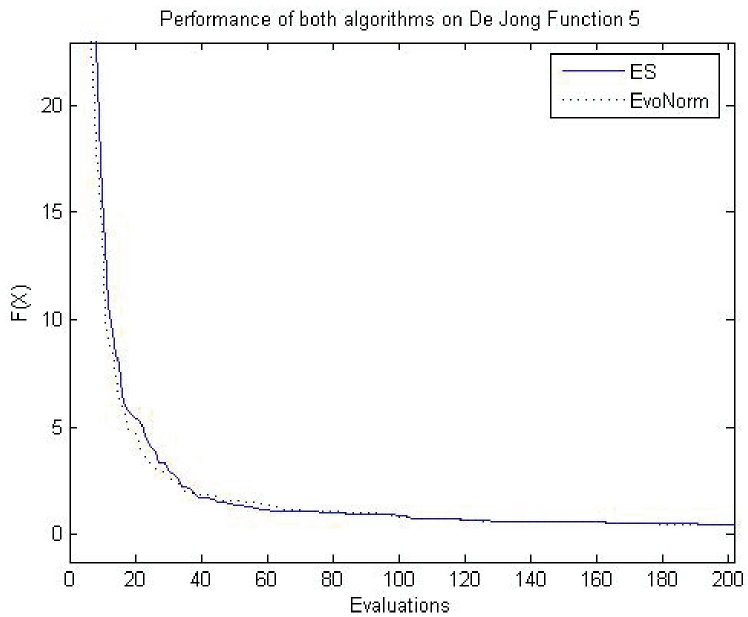

Figure 5. Performance of the algorithms to optimize De Jong function 5.

4 Conclusions

The performance of Evonorm is superior to evolution strategies in De Jong functions 1, 3 and 4. In other functions 2 and 5 the performance is very similar in both algorithms. It is important to mention the use of exploration in these test functions. There is an open opportunity to improve this algorithm because it is necessary to test Evonorm with and without exploration. The Evonorm is effective to optimize De Jong functions and the implementation of the algorithm is easy because the calculation of mean and standard deviation involve common used arithmetic operations. Evonorm is an evolutionary algorithm for continuous optimization based in estimation of parameters of random variables with normal distribution functions. The parameters are calculated from a set of selected individuals. The algorithm shows a good performance with the comparison again Evolution Strategies. The future work includes new test functions and comparisons with similar evolutionary algorithms for continuous optimization and makes the algorithm more independent to the problem. It is supposed an independent interaction between variables so will be important to may use multivariable normal distribution functions and different distribution functions not only the normal one. It is expected to extend the use of Evonorm in multimodal, constrain satisfaction and multi objective optimization.

References

1. Baldonado, M., Chang, C.-C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. Int. J. Digit. Libr. 1 (1997) 108–121
2. Bruce, K.B., Cardelli, L., Pierce, B.C.: Comparing Object Encodings. In: Abadi, M., Ito, T. (eds.): Theoretical Aspects of Computer Software. Lecture Notes in Computer Science, Vol. 1281. Springer-Verlag, Berlin Heidelberg New York (1997) 415–438
3. van Leeuwen, J. (ed.): Computer Science Today. Recent Trends and Developments. Lecture Notes in Computer Science, Vol. 1000. Springer-Verlag, Berlin Heidelberg New York (1995)
4. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)
5. Shumeet Baluja, 1994, “Population-Based Incremental Learning: {A} Method for Integrating Genetic Search Based Function Optimization and Competitive Learning”, Tech. Rep. No. CMU-CS-94-163, Pittsburgh, PA, Carnegie Mellon University. (1994)
6. H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions I", In "Parallel Problem Solving from Nature", 178-187, Lecture Notes in Computer Science", September (1996)
7. Georges R. Harik and Fernando G. Lobo and David E. Goldberg, “The compact genetic algorithm", IEEE transactions on evolutionay computation. Volumen 3, NUMBER 4, 287-297, November. (1999)
8. P. Larrañaga, R. Etxeberria, J.A. Lozano, and J.M. Peña, “Optimization in continuous domains by learning and simulation of gaussian networks.”

9. In A.S. Wu, editor, Proceeding of the 2000 Genetic and Evolutionary Computation Conference Workshop Program, (2000) pages 201-204
10. Peter Alexander Nicolaas Bosman, "Design and Application of Iterated Density Estimation Evolutionary Algorithms" Utrecht, Institute of Information and Computing Sciences. Phd. Thesis. (2003)

11. Martin Pelikan and David E. Goldberg and Erick Cantú-Paz, "BOA: The Bayesian Optimization Algorithm", Wolfgang Banzhaf and Jason Daida and Agoston E. Eiben and Max H. Garzon and Vasant Honavar and Mark Jakiela and Robert E. Smith, "Proceedings of the Genetic and Evolutionary Computation Conference {GECCO}-99, 525--532", Orlando, FL, Morgan Kaufmann Publishers, San Francisco, CA. (1999)
12. Martin Pelikan and Kumara Sastry and Martin V. Butz and David E. Goldberg, "Hierarchical BOA on Random Decomposable Problems", Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign., IlliGAL Report No. 2006002, January, (2006)
13. Cláudio F. Lima and Kumara Sastry and David E. Goldberg and Fernando G. Lobo, "Combining Competent Crossover and Mutation Operators: A Probabilistic Model Building Approach", Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign. IlliGAL Report No. 2005002}, February, (2005)
14. P. Larrañaga and J. A. Lozano, "Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation." Kluwer Academic Publishers (2001)
15. Cantú-Paz, E., "Feature subset selection by estimation of distribution algorithms". In W. B. Langdon, E. Cantu-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, N. Jonoska (Eds.), GECCO-2002: Proceedings of the Genetic and Evolutionary Computation Conference. (pp. 754). San Francisco, CA: Morgan Kaufmann. (2002)
16. Thomas Bäck", "Evolutionary Algorithms in Theory and Practice", Oxford University Press, New York, (1996)
17. K. A. DeJong, "An analysis of the behavior of a class of genetic adaptive systems", journal= "Dissertation Abstracts International, 36(19), 5140B, University Microfilms No. 76-9381. (1975)
18. Zbigniew Michalewicz, "Genetic algorithms + data structures = evolution programs", Springer-Verlag", New York. (1999)