EvoNorm, a new evolutionary algorithm to continuous optimization

Luis Torres-T
Corporación Mexicana de Investigación en Materiales, S.A.
Oceanía 190
25290 Saltillo, Coah. México
ltorres@comimsa.com.mx

ABSTRACT

This paper shows the first results of a new evolutionary algorithm which the population is built by normal distribution functions. Computer simulation compare Evonorm versus Evolution strategies to optimize the Goldstein - Price function. The results show an advantage of EvoNorm as evolutionary algorithm to optimize functions where the interactions between variables is weak.

Keywords

Evolutionary algorithms, estimation of distribution algorithms, probabilistic model-building genetic algorithms, continuous optimization, evolution strategies

1. INTRODUCTION

There is a new tendency to generate simplified versions of evolutionary algorithms where crossover and mutation procedures are eliminated. The population is built by a model that represents an estimation of distributions where the model parameters are defined by selected individuals. Examples of these algorithms are the Population-Based Incremental Learning ([1]), the Compact Genetic Algorithm ([3]), the Bayesian Optimization Algorithm [7] and the Univariate Marginal Distribution Algorithm ([6]). These algorithms consider a weak interaction between variables. The only exception is the Bayesian Optimization Algorithm. In recent years has been appearing new algorithms where consider a strong interaction between variables like Hierarchical Bayesian Optimization Algorithm ([8]) and the Extended Compact Genetic Algorithms ([4])

Evonorm is a new evolutionary algorithm where the population is built by normal distribution functions. The parameters of these normal distribution functions 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 ones to calculate parameters of distribution functions and to generate new individuals from these distribution functions.

The present section is the introduction. The second section is a review of the EvoNorm. The third section shows the optimization of a test function with EvoNorm versus Evolution strategies (μ, λ) to make comparison between them. The conclusion and future work is given in section forth.

2. EVOLUTIONARY ALGORITHM BASED IN NORMAL DISTRIBUTION FUNCTIONS

The normal distribution function is a random variable and describes many random phenomena that occurs in every day life. The normal distribution function has 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.

The EvoNorm procedure has the same philosophy of an evolutionary algorithm, 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. EvoNorm evolves the parameter of every normal distribution function to generate new real vectors of decision variables that will be evaluated and some of them selected to calculate new parameters of the distribution function to generate a new population. The process is repeated again and again as it is shown in the general algorithm of the Table 1.

There are two visible parameter of the algorithm. The first parameter is the number of individuals generated per iteration M , and the second parameter is the number of individuals selected per iteration N where it must be a percent of the total population; in other words, N represents a percent of M like 10 or 20 percent. There are another two practical parameter not described here but they will be discussed in the next section.

3. EXPERIMENTAL RESULTS

The direct use of the parameters calculated from the indi-

Table 1: General evolutionary algorithm of normal distribution functions.

1) Generation of a population of size M with uniform distribution functions. 2) Evaluation of the population. 3) Selection of the best N individuals where $N << M$. 4) Calculation of the mean and standard deviation with N selected individuals of step (3) 5) Generation of a new population with normal distribution functions. The mean and the standard deviation of these functions were calculated in step (4) 6) If a condition of conclusion is not satisfied go to step (2) otherwise end. EvoNorm parameters M: number of individuals generated per iteration N: number of individuals selected per iteration

viduals selected causes a poor performance of the algorithm. It was necessary to make some changes to the general algorithm to make it practical. The mean is used directly but a value of 0.5 in the standard deviation is used in a 50 percent of total iterations. The value of 0.5 was acquired by trail and error. The value of the standard deviation calculated from the selected population is taken directly when there is more than 50% of total iterations.

The exploration of the algorithm is high at the half of the total iterations. The other half of the algorithm reduces the exploration because the standard deviation decrease drastically. The following condition is added in step (4) to the general algorithm of EvoNorm.

If actual generation $<$ 50% of total iterations then Standard deviation of all the normal distribution function is equal to 0.5 Else Standard deviation is calculated from a population of individuals selected.

There are two new parameter of the algorithm, the percent of the total generation to consider another standard deviation value and the standard deviation value to be used by the normal distribution functions. In the experiments it was used percent of 50% and a standard deviation of 0.5 as mention above. The algorithm was tested to optimize the Goldstein - Price function [5]:

$$\begin{aligned} a &= 1+(x_1+x_2+1)^2 \\ b &= (19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2) \\ c &= 30+(2x_1-3x_2)^2 \\ d &= (18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2) \\ f(x_1,x_2) &= abcd \end{aligned}$$

where $-2 <= x_i <= 2$. The function has a global minimum value of 3 at $(x_1, x_2) = (0, -1)$

It was used an evolution strategies (μ, λ) with self adaptation without recombination [2] to optimize the same function to make a comparisons between EvoNorm and evolution strategies (ES). The parameters of both algorithms like the total of individuals and the total of iterations are adjusted

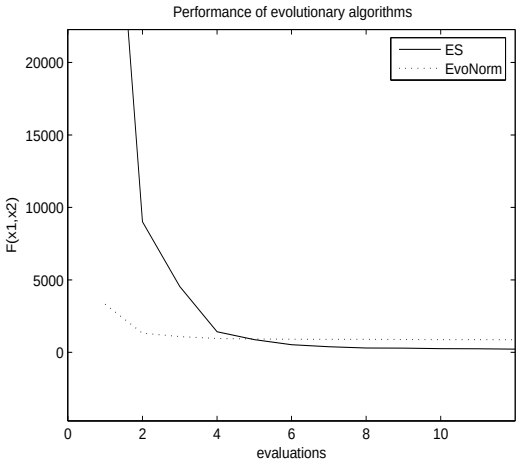

Figure 1: A zoom of the first 20 evaluations.

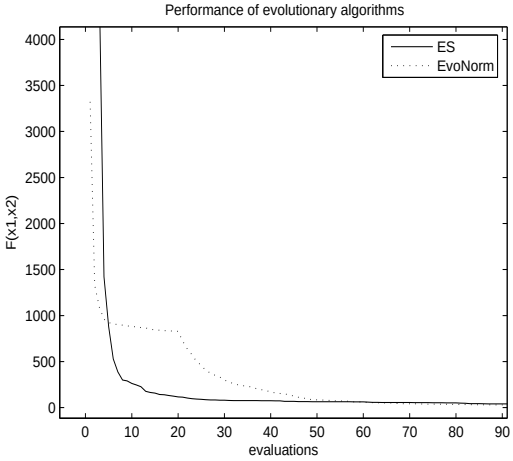

Figure 2: A zoom between 1 and 100 evaluations .

to get 400 evaluations. ES is $(20, 20)$ and runs in 20 iterations and EvoNorm has $M = 20$ with $N = 5$ and runs in 20 iterations. Both algorithms were executed 100 times to get an average of the performance. The figure 1 shows the average performance of 100 runs in the first 20 evaluations. ES has a slighter better performance than EvoNorm between evaluation 4 and 15 but ES has a poor beginning than EvoNorm between evaluation 1 and 4. The figure 2 shows the average performance of both algorithms in the first 120 evaluations. The ES decrease smoothly to the optimum and EvoNorm decreace abruptly between evaluation 20 and 60; it behaves like evolution strategies later. The figure 3 shows the performance of both algorithms in the last 300 evaluations. EvoNorm has better performance than ES because EvoNorm reachs the optimum more times than ES (in 100 runs EvoNorm reaches 59 times a value smaller than 3.1 whereas ES reaches it only 5 times)

4. CONCLUSIONS

The EvoNorm is a new evolutionary algorithm for continuous optimization based in a estimation of parameters of

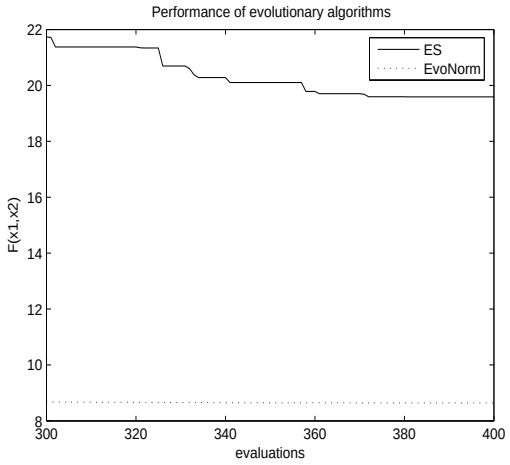

Figure 3: A zoom of the last 100 evaluations.

normal distribution functions. The estimation is calculated from a set of selected individuals. The algorithm shows a good performance with the comparison again another evolutionary algorithm as evolution strategies. The future work includes new test functions and comparisons with similar evolutionary algorithms for continuous optimization. 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.

5. REFERENCES

[1] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Tech. Rep. No. CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, PA, 1994.

[2] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, 1996.

[3] G. R. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. *IEEE transactions on evolutionay computation.*, 3(4):287–297, November 1999.

[4] C. F. Lima, K. Sastry, D. E. Goldberg, and F. G. Lobo. Combining competent crossover and mutation operators: A probabilistic model building approach. Technical Report IlliGAL Report No. 2005002, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign., February 2005.

[5] Z. Michalewicz. *Genetic algorithms + data structures = evolution programs*. Springer-Verlag, New York, 1999.

[6] H. Muhlenbein and G. Paa. From recombination of genes to the estimation of distributions i. In *Parallel Problem Solving from Nature*, pages 178–187. Lecture Notes in Computer Science, September 1996.

[7] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian optimization algorithm. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, pages 525–532, Orlando, FL, 1999. Morgan Kaufmann Publishers, San Francisco, CA.

[8] M. Pelikan, K. Sastry, M. V. Butz, and D. E. Goldberg. Hierarchical boa on random decomposable problems. Technical Report IlliGAL Report No. 2006002, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign., January 2006.