STRATEGIES FOR AVOIDING LOCAL EXTREME IN FUNCTION OPTIMIZATION THROUGH BACTERIAL CHEMOTAXIS

H. Alvarez, O. Quintero, A. Angel and M. Yepes

Escuela de Procesos y Energía. Facultad de Minas, Universidad Nacional de Colombia.
Grupo de Automática de la Universidad Nacional (GAUNAL).
Carrera 80, barrio Robledo. Medellín, Colombia.
e-mail: hdalvare @ unalmed.edu.co

Abstract: In this work the basic Bacterial Chemotaxis (BC) algorithm is presented as a tool for optimizing the typical non linear-non convex functions that appear during advance control strategies implementation. Two strategies are proposed for avoiding the local extreme stopping of the basic BC algorithm. The basic BC algorithm and its two improving proposals are tested using three reported benchmark functions for optimization algorithms and one own function stated for representing typical control situations. Results are compared with the original BC algorithm.

Keywords Optimization, Bacterial Chemotaxis, Searching procedures.

1. INTRODUCTION

The majority of advanced strategies for modeling, identification and control require almost a numerical optimization of multivariable nonlinear functions. The problem in this class of optimization may be formulated as to found a vector θ of parameters, which produces a minimum (or maximum) value of a prescribed function (cost or fitness function). In optimal control and model predictive control, a non convex optimization problem is frequently found and an efficient optimization technique is required (Menold et. al., 1996; Alvarez, 2000). Unfortunately, there is not a robust and efficient technique that resolve such an optimization problems. Therefore, some of the last optimal and predictive control algorithms use objective function reduction in order to obtain a convex optimization problem (Espinosa, 2001). Under these considerations it results very interesting to explore proposals that allow resolve the non-convex optimization problems that appear in optimal control and predictive control applications.

Until now, the spectrum of tools for executing this optimization is wide. The aim of the present work is to present the BC optimization algorithm as a usefulness tool when it is used as a part of advanced control strategies. Additionally it is showed that the original BC algorithm must be improved in order to escape from local extreme. Two strategies are proposed for achieving this objective: a summation interval modification and a parallel search method.

2. FUNCTION OPTIMIZATION THROUGH BACTERIAL CHEMOTAXIS ALGORITHM

The Bacterial Chemotaxis (BC) algorithm was originally proposed in (Bremmerman and Anderson, 1990) as a numerical optimization of the weights in a feedforward artificial neural network. The BC searching mechanism imitates the typical bacteria colony movements inside a cultivation broth. The bacteria colony movements are a direct function of the nutrient concentration in the cultivation broth. Such functionality is composed of a sort of bacteria communication code, which provides information about nutrient concentration in various directions around the current bacteria colony position. The final colony movement in the next time step will be in that direction of increased nutrient concentration (Spiro et al., 1997; Kearns and Shimkets, 1998). The same criterion of movement is applied in the numerical searching for a maximum (or minimum) of any multivariable function.

In the majority of engineering problems it is possible to formulate a Fitness Function (***FF***), which evaluates the advantages of a problem solution. Such a problem solution is currently stated as a group of parameters, in the next namely Optimal Parameter vector θ_{OPT}. Then, and without loss of generality, in the sequel a minimum function extreme is considered as the wanted optimum

function value. This minimum value corresponding to the most adjusted problem solution found in accordance to the ***FF*** definition. Therefore ***FF*** will converge to a maximum, which corresponding to an inverse of the value of the optimized function. In a first step of the BC algorithm, the nominal interval of the parameter $\boldsymbol{\theta}$ to be optimized are determined. A seed value is stated for each parameter θ_i. Then, the BC algorithm begins its work by exploring some directions in the parameter space. The basic BC algorithm may be described as the next sequence of steps for a single bacterium or a bacteria colony:

1. A nominal value for each one of the ***N*** parameter is determined and taken as the provisional Optimal Parameter vector $\boldsymbol{\theta}_{OPT}$. A Fitness Function value for function $\boldsymbol{f}(\boldsymbol{\theta})$ to be optimized is calculated and taken as a provisional optimal function value (FF_{OPT}).
2. Taking into account the interval of each parameter (θ_{imin}, θ_{iMax}), a summation vector $\mathbf{S}_i = [s_1\ s_2\ \ldots\ s_N]^T$, of terms (one for each parameter) is randomly determined inside an interval $(0,\pm\Delta\theta_i)$ for each s_i. This procedure allows positive or negative values in the $\mathbf{S}_i$ vector.
3. A new parameter vector $\boldsymbol{\theta}_i$ is calculated using the previously found summation vector $\mathbf{S}_i$ as: $\boldsymbol{\theta}_i = \boldsymbol{\theta}_{OPT} + \mathbf{S}_i$.
4. Using the new parameter vector $\boldsymbol{\theta}_i$ a Fitness Function (FF_i) value for the optimized function $\boldsymbol{f}(\boldsymbol{\theta})$ is determined.
5. If $FF_i > FF_{OPT}$ then the set of parameters $\boldsymbol{\theta}_i$ is stored as the new $\boldsymbol{\theta}_{OPT}$ and the current summation vector is maintained. If $FF_i < FF_{OPT}$ then the original $\boldsymbol{\theta}_{OPT}$ is maintained and a new random summation vector is determined.
6. If the finishing condition is not fulfilled the algorithm returns to step 3. Else, the algorithm ends and the answer is $\boldsymbol{\theta}_{OPT}$.

In (Bremermann and Anderson, 1990) the use of a Gaussian distribution with mean $\mu=0$ and standard deviation $\sigma=1.0$ is justified to generate random summation vectors. These authors justified the use of this class of probability distribution based on Gaussian fluctuations of neuronal connections in human brain. Taken into account that basic BC algorithm was stated for artificial neural networks training, the use of Gaussian distribution is justified. However, it is possible to use other class of probability distribution for randomly select summation vectors. In the present work, an equal probability distribution was successfully used.

There exist three possible finishing conditions. The first one takes into account a fixed number of algorithm cycles. A second condition looks for a quasi-invariant ***FF*** value during a fixed number of algorithm cycles. The third possible finishing condition verifies if a pre-stated desired ***FF*** value is reached. As will be seen later, the proposed modifications to the basic BC algorithm presented here uses the second finishing condition. Additionally, the proximity of the current maximum to others possible extremes in the parameter space is tested. In accordance with this evaluation, each modification adds an escape policy to the basic BC algorithm.

3. TWO STRATEGIES FOR IMPROVING THE BC OPTIMIZATION

The two proposed modifications are intended looking for a mechanism that allows escape from local function extreme (in this case local extreme of ***FF***). It is known that the best knowledge about parameter space produces the best optimization result. Therefore, the modifications proposed here will operate provided an acceptable knowledge about parameter interval be available.

3.1. Summation interval modification

In the finishing test, this first proposed BC algorithm modification verifies when a previously found $\boldsymbol{FF}_{MAX}$ is not enhanced during a number of algorithm executions. Then, a progressive enlargement of the range of the summation vector $\mathbf{S_i}$ is done through a $\Delta\boldsymbol{\theta}_I$ expansion vector. Such an increment is implemented until reaching a new $\boldsymbol{FF}_{MAX}$ (function extreme) or reaching one of the limits in the interval stated for the $\mathbf{S_i}$ vector. Therefore, the second possible finishing condition of the previously presented basic BC algorithm is reformulated. This finishing condition resets to 1 a counter of $\boldsymbol{FF}_{MAX}$ occurrences ($\mathbf{C}_{FFMAX}$) when a new extreme function is found. $\mathbf{C}_{FFMAX}$ is increased if the group of parameters $\boldsymbol{\theta}_i$ produces a ***FF*** equal to the previously found ***FF***. When $\mathbf{C}_{FFMAX}$ reaches a threshold value, the summation term interval $(0, \pm\Delta\boldsymbol{\theta}_i)$ for each parameter is increased using $\Delta\boldsymbol{\theta}_i = \Delta\boldsymbol{\theta}_i + \delta * \Delta\boldsymbol{\theta}_i$. If the new summation interval

exceeds the nominal parameter interval, then the optimization process finish and the last found optimum is taken as the answer. Else, the new summation term interval is maintained, $\mathbf{C}_{FFMAX}$ is reset to 1.0 and the algorithm is repeated from step 2 (see algorithm sequence in section 2.). It is important in this point to mention that a narrow parameter interval (problem constraints) will convert a local maximum in an global one. Therefore, the pretended algorithm modification for relative extremes escape will result unnecessary.

As can be seen in Figure 1, using this modification, every time the BC algorithm found a function extreme value, the exploration policy is modified in other to looking for other function extremes in an expanded neighborhood to the current point. This neighborhood expansion finishes when the summation term interval reaches the border of parameter space or finds a new function extreme.

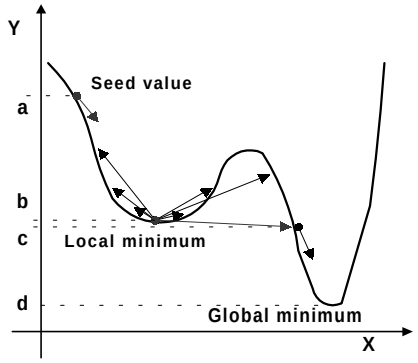

Figure 1. An illustration of the operation of first algorithm modification.

In regarding to this topic, it is possible that the current parametrization of the ***FF*** lets open one side of the interval of parameter vector. Such a situation is commonly found in many engineering problems, for example in chemical processes, when there are two or more operational points and a wide parameter expansion is possible (Menold et al., 1996; Alvarez, 2000). Under this condition, the step size results a crucial operation value, which may produce erroneous algorithm termination. Therefore, the previously mentioned finishing condition will not be useful. An alternative proposal is to stop the neighborhood expansion when an appropriate ***FF*** value is reached (sub-optimal value) or when the BC algorithm completes a number of iterations.

3.2. Parallel searching modification

The other proposed modification to the algorithm operation is to maintain the searching method but using a set of bacteria instead of a single cell or colony. In this proposal, the colony is separated in order to have available a group of individual bacteria, each one looking for the optimum of the function without taking into account other bacteria performance. The algorithm begins with a random distribution of the bacteria in the problem space. Next, each bacterium performs a searching for the maximum using the original BC algorithm. The process ends when a fixed number of steps have been executed or when during a consecutive number of steps a group of bacteria remain at the same maximum position. It must be noted that if multiple local extremes there exist, after a time, a number of bacteria will stay in local extremes but at least one of them will end at the global extreme. This BC algorithm modification has the local extreme escape as its desirable property but its computational burden is bigger than the first modification previously presented.

In order to reduce such a high computational cost some alternatives are possible. The alternative proposed here is to add the original BC algorithm with a counter of $\boldsymbol{FF}_{MAX}$ occurrences ($\mathbf{C}_{FFMAX}$) when a new extreme function is found (like was proposed above). In this way, during any bacterium movements a test on $\mathbf{C}_{FFMAX}$ can be performed. When a bacterium reaches a local minimum, the $\mathbf{C}_{FFMAX}$ will reaches a threshold value and the algorithm excludes this bacterium of the bacteria set. Therefore, only the promising bacteria will remain in the set performing the algorithm steps and the computational burden will be reduced with the algorithm progress. In Figure 2., the parallel search modification is illustrated with a colony of four bacterium, a very low number of bacterium but enough for this simple function in R^2. As can be seen, bacteria initially located at seeds *a* and *c* will fall into a local minimum. On the other hand, bacteria located in seed *b* or *d* will reach the global minimum. The number or steps that each bacterium executes before end will be related to the summation interval (fixed), to the seed value for the parameters and to the function gradient in the seed zone.

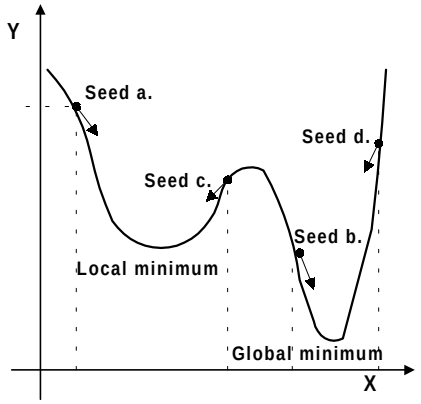

Figure 2. An illustration of the operation of second algorithm modification.

4. EXAMPLES OF OPTIMIZATION USING BASIC AND MODIFIED BC

In this section, four optimization problems are presented using the basic and the two BC algorithms modifications. A comparison is done in order to demonstrate the effectiveness of algorithm modified versions, when the optimization fails using the basic BC. The four mathematical functions were selected looking for an easy understanding of optimization procedure. Additionally, two of the selected functions have the two main particularities of the majority of optimization problems: limited parameters interval and multiple local extremes. For all the functions, the parameter vector is: $\boldsymbol{\theta}$ = [x, y]. Three functions were taken from (Whitley et al., 1985), frequently referred as benchmarks for numerical optimization. Additionally, a own function was generated looking for a more realistic problem, as during control policy optimization: several local extremes but smooth function behavior.

4.1 Optimization of Griewangk's Function:

This is a function with a soft behavior in the domain [-100, 100]x[-100, 100] and with only one absolute extreme, as may be seen in Figure 3. The formulation of Griewangk's function is:

$$z = 1 + \left(\frac{1}{200}\right)\cdot\left(x^2 + y^2\right) - \cos(x)\cdot\cos\left(\frac{y}{\sqrt{2}}\right)$$

This function has its absolute minimum at [0, 0] with z=1. By using the basic BC algorithm the global minimum was found easily for an individual bacteria in around 23 iterations. Therefore, this benchmark problem is not a real challenge for the basic BC and improvements to this algorithm will result unnecessary.

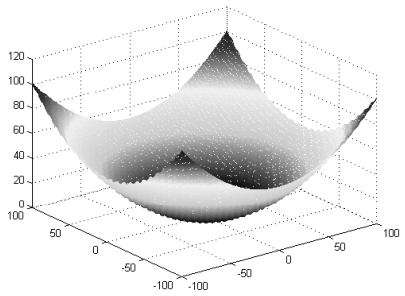

Figure 3. Griewangk's function in 3D.

4.2 Optimization of Modified Rosembrock's Function:

This function, a modified Rosembrock's banana, has a soft behavior in the domain [-2, 2]x[-2, 2], with a local minimum in (1, 1) and an global minimum in (-1, -1). In Figure 4, a 3D plot is presented.

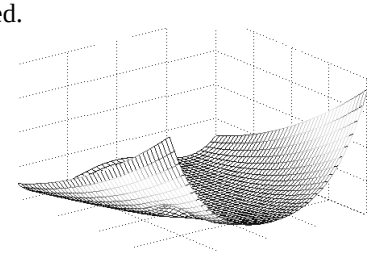
Figure 4. Modified Rosembrock´s function in 3D.

The Modified Rosembrock's function is:

$$z = 74 + 100\cdot\left(y - x^2\right)^2 + (1-x)^2 + 400\cdot e^{\left(-\left(\frac{(x+1)^2+(y+1)^2}{0.1}\right)\right)}$$

Using the basic BC algorithm it could be observed that the global minimum was found, but some tests stopped in the local minimum. By using the first modification, to find the global minimum was so easy for an individual bacteria in around 35 iterations. Under this consideration, the first improving strategy is enough to optimizing such a function. Tests with the second strategy showed that the second strategy is also useful for finding the global minimum, but its computational burden is a drawback that inclines the choice to the first strategy.

4.3 Optimization of Rastrigin's Function:

This function has several local extremes in the domain [-5.12, 5.12]x[-5.12, 5.12] and a big gradient as may be seen in Figure 5. The formulation of Rastrigin's function is:

$$z = 20 + \left(x^2 - 10 \cdot \cos\cdot\left(2 \cdot \pi \cdot x\right)\right) + \left(y^2 - 10 \cdot \cos\cdot\left(2 \cdot \pi \cdot y\right)\right)$$

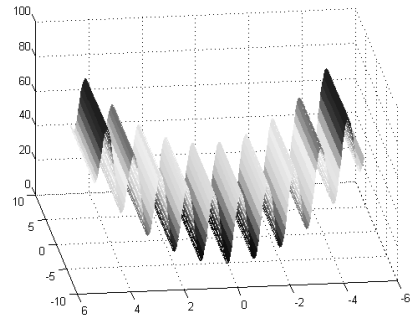

Figure 5. Rastrigin's function in 3D.

Due to the high gradient of this function, the basic BC algorithm can not escape from local extremes. Therefore, it is necessary to implement an improvement to the basic algorithm in order to avoiding local extremes. Although for this function the two proposed strategies may be applied with success, a difficult tuning task over algorithm parameters must be done when first proposal is applied. It must be noted that all the procedure assumes that function form is unknown (this topic will be reviewed in conclusion section). Therefore, only the second proposal was applied with good results. In this case, around 12 movements were executed by a bacterium before it found the absolute minimum. It must be noted that the random distribution of the seed should guarantee an uniform distribution in the function space in order to cover all the function extreme vicinities.

4.4 Optimization of Own Function :

The last function was generated for the authors looking for a complex behavior but maintaining a relative simple mathematical formulation. Additionally, as it was mentioned before, a more realistic function in regarding to optimization problems found in control policy determination is preferable. In this section an extended discussion about basic and improved BC algorithm used for optimizing such a function is given. In Figure 6, this mathematical function is plotted. The function formulation is:

$$z = 3.5 + (0.025*(3.5 - y))^2 + \cos(x) - \sin(xy)$$

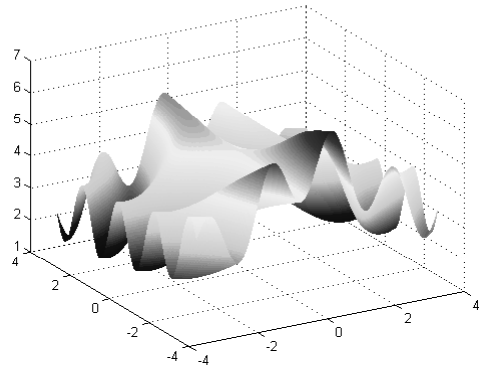

Figure 6. Own function in 3D.

In the space [-3.5, +3.5]x[-3.5, +3.5] this function presents eight minima but only one is a global minimum, located at [-3.1416, 3.5] with a value of z = 1.5000. The optimization was firstly done using the basic BC algorithm with three seed parameters $[x, y]_{Seed}$: one randomly selected [-2.365, 1.014], other fixed in space origin [0, 0] and the third near to global minimum [-3, 3]. In all cases, the summation vector was restricted to the fixed interval $\Delta\boldsymbol{\theta} \in$ [-0.05, 0.05] and 15000 algorithm iterations. With a random seed, the bacterium executes 75 success movements before reach a local minimum at z=1.5983. When origin was used as seed, the bacterium executes 296 success movements before reach another local minimum at z=1.7235. Finally, with the seed near to global minimum only 30 success movements were necessaries to arrive to global minimum at z=1.500. Additionally, the algorithm was registering the repeated minimum values and just stopped when this number reaches a threshold value of 10.

a. Performance of the summation interval modification

In accordance with the previously discussed algorithm modifications, in this part the first modified algorithm is tested. The optimization was performed with a single bacterium with the three seeds used in the previous basic BC test. The algorithm was stopped when the current minima was unchanged during 20 bacterium movements provided summation interval has its maximum value. The initial $\Delta\boldsymbol{\theta}_i$ was the used in the basic BC

test $\Delta\boldsymbol{\theta} \in [-0.05, 0.05]$ and the increment parameter was fixed in $\delta = 0.2$. For all the cases the bacterium found the global minimum. The bacterium movements where 12536 for random seed, 8597 for the seed in the origin and 1825 for the seed near to global minimum.

b. Performance of the parallel searching modification

In this case the optimization was performed using the parallel searching modification. The number of algorithm iteration was fixed in 15000 for each bacterium and the summation vector was restricted to the fixed interval $\Delta\boldsymbol{\theta} \in [-0.05, 0.05]$. Ten bacteria were randomly distributed in the problem space and the algorithm was executed. In spite of its bigger computational burden respecting to the original algorithm, the parallel modification found in all cases the global function minimum. Additionally, using the individual bacterium stopping procedure, the number of bacteria movements was reduced from 150000 (=10 bacteria*15000 movements per bacterium), to 44362. This fact represent a reduction of. 70.33% of expected bacteria movements. Regardless of the low bacterium movements of the basic BC algorithms (15000), the basic algorithms demonstrated its low performance when there are several local minima in the function. Therefore, the increment in computational cost from 15000 to about 45000 movements is justified taken into account the success of the modified BC algorithm. Another topic to be noted in the above test is the fast convergence of the bacterium that begins with seed near to the global minimum. This fact emerges from the high slope of the function near to global minimum. Additionally, the other bacteria fall in very plane function valleys, therefore they need more movements to arrive to their local extreme.

5. CONCLUSIONS

Two modifications to original BC algorithms were proposed and tested in this work. The results showed that both algorithm modification allow escape of local extreme of the function being optimized. This fact permits to use BC algorithm in optimizing high nonlinear functions and guarantees to find a global optimum. Unfortunately, both proposal increment the computational burden regard the basic BC algorithm. Therefore it is necessary to consider such a disadvantage when the modified BC algorithm be implemented. However, it is possible to reduce the computational burden by means of a previously knowledge about the function parameters space and parameters restrictions. In this way an enhanced seed location for the BC algorithm helps to obtain the global optimum with a few bacterium steps (first proposed modification) or with some bacteria in a separate colony (second proposed modification).

There exist a lot of work to be done in order to find a optimum way of locate parameter seeds, given that such a task is capital in the BC success. About this topic there are several works that present procedures and algorithm for selecting seeds in iterative optimization methods. Unfortunately, each function to be optimized requires a particular situation study and no general method can be given. It will be interesting to explore some seed distribution strategies coupled with the two BC modifications presented in this work.

REFERENCES

Alvarez, H. (2000) Model predictive control based on fuzzy model for the pH control. (*In Spanish*). PhD thesis. INAUT-UNSJ. Argentina.

Bremermann, H. and Anderson, R. (1990) An alternative to back-propagation: a simple rule of synaptic modification for neural net training and memory. Technical Report PAM-483. Center for pure and applied mathematics. University of California, Berkeley.

Espinosa, J. (2001). Fuzzy modeling and control. PhD thesis. Katholieke Universiteit Leuven, Belgium.

Kearns, D. and Shimkets, L. (1998). Chemotaxis in a glinding bacterium. Proc. Natl. Acad. Sci. USA. Vol. 95, pp 11957-11962.

Menold, P.; Allgower, F. and Pearson, R. (1996) Nonlinear structure identification or chemical processes. Technical Report AUT96-32. Insitut fur Automatik-Zurich.

Spiro, P.; Parkinson, J. and Othmer, H. (1997) A model of excitation and adaptation in bacterial chemotaxis. Proc. Natl. Acad. Sci. USA. Vol. 94, pp 7263-7268.

Whitley et al. (1985). Building better test functions. Proc. of the 6th Int. Conf. on GAs. Morgan Kaufman.